DOCUMENT RESUME

ED 115 258 IR 002 802

AUTHOR Bellamy, Harold J.; And Others
TITLE Design of Training Systems. Program Maintenance Manual; Data Base, ETE, SCRR, and TPF Models. Report No. 29.
INSTITUTION Naval Training Equipment Center, Orlando, Fla. Training Analysis and Evaluation Group.
REPORT NO TAEG-R-29
PUB DATE Sep 75
NOTE 617p.; For a related document see IR 002 803; Not available in hard copy, due to print size of the original document

EDRS PRICE MF-$1.08 Plus Postage. HC Not Available from EDRS.
DESCRIPTORS *Computer Programs; Data Bases; Educational Technology; Flow Charts; *Instructional Design; Job Analysis; *Job Training; *Management Systems; Military Training; Models; Occupational Information; On Line Systems; Program Descriptions; Skill Analysis; *Systems Approach; Time Sharing; Training; Training Objectives
IDENTIFIERS Program Documentation; *Training Analysis and Evaluation Group

ABSTRACT

This report contains detailed information on the three Design of Training Systems (DOTS) models and the DOTS data base. It consists of a single volume containing a description and macro flow, detailed logic flows and program listings for the Educational Technology Evaluation (ETE) model, the System Capabilities Requirements and Resources (SCRR) model, the Training Processes Flow (TPF) model, and the DOTS data base. Control logic, input/output record formats, the temporary and permanent data files are described for each subsystem (the three models and the data base). The information contained in this volume is intended for use by programmers given the task of installing and modifying the DOTS programs. (Author/EMH)

****
* Documents acquired by ERIC include many informal unpublished *
* materials not available from other sources. ERIC makes every effort *
* to obtain the best copy available. Nevertheless, items of marginal *
* reproducibility are often encountered and this affects the quality *
* of the microfiche and hardcopy reproductions ERIC makes available *
* via the ERIC Document Reproduction Service (EDRS). EDRS is not *
* responsible for the quality of the original document. Reproductions *
* supplied by EDRS are the best that can be made from the original. *
****

TAEG

TRAINING
ANALYSIS
AND
EVALUATION
GROUP

TAEG REPORT
NO. 29

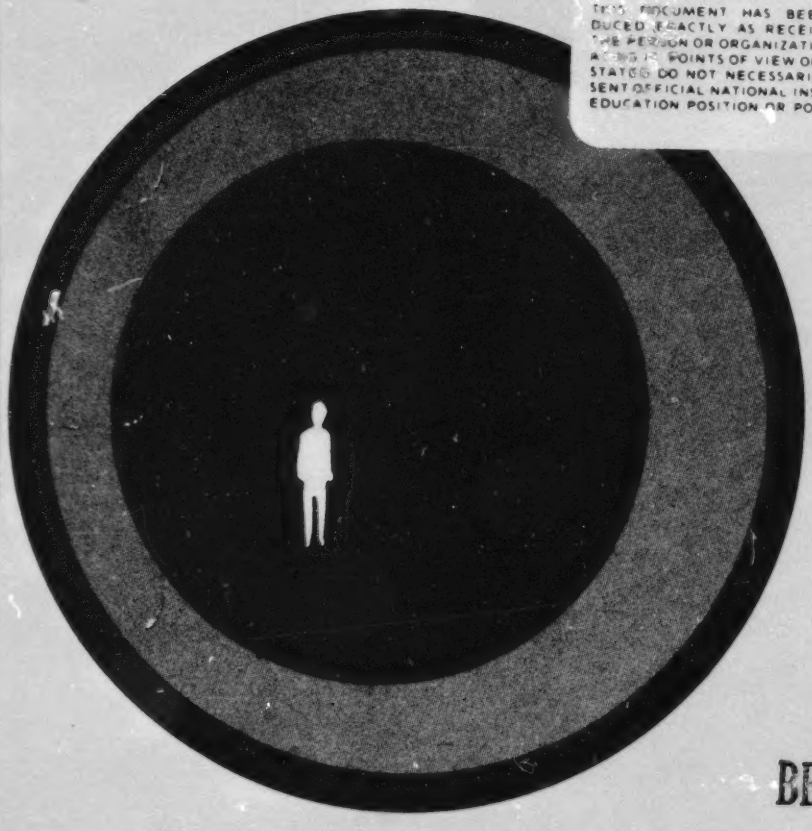
DESIGN OF TRAINING SYSTEMS PROGRAM MAINTENANCE MANUAL

DATA BASE, ETE, SCRR, AND TPF MODELS

ED115258

U S DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY



FOCUS ON THE TRAINED MAN

BEST COPY AVAILABLE

2

SEPTEMBER 1975

IR002802

TRAINING ANALYSIS AND EVALUATION GROUP
ORLANDO, FLORIDA 32813

Technical Report: TAEG REPORT NO. 29

DESIGN OF TRAINING SYSTEMS

PROGRAM MAINTENANCE MANUAL

ABSTRACT

This report contains detailed information on the three Design of Training Systems (DOTS) models and the DOTS data base. It consists of a single volume containing a description and macro flow, detailed logic flows and program listings for the Educational Technology Evaluation model (ETE), the System Capabilities Requirements and Resources (SCRR) model, the Training Process Flow (TPF) model and the DOTS data base.

Control logic, input/output record formats, and temporary and permanent data files are described for each subsystem (the three models and the data base). The information contained in this volume is intended for use by programmers tasked with installing or modifying the DOTS programs.

The programs described in this manual were developed by IBM Corporation for the Training Analysis and Evaluation Group, Orlando, Florida (Contract No. N61339-73-C-0097).

Reproduction of this publication in whole or in part is permitted for any purpose of the United States Government.

TAEG Report No. 29

DESIGN OF TRAINING SYSTEMS

PROGRAM MAINTENANCE MANUAL

DATA BASE, ETE, SCRR, AND TPF MODELS

Harold J. Bellamy
Kenneth V. Branch
James D. Staley
Ronald E. Yanko

This Report Was Prepared

by

International Business Machines Corporation

for the

Training Analysis and Evaluation Group

September 1975

BEST COPY AVAILABLE

Alfred F. Smode

ALFRED F. SMODE, Ph.D., Director,
Training Analysis and Evaluation Group

U S DEPARTMENT OF HEALTH, EDUCATION & WELFARE
NATIONAL INSTITUTE OF EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

TAEG REPORT NO. 29

TABLE OF CONTENTS

SECTION		PAGE
I	INTRODUCTION	
	PURPOSE	I-1
	ORGANIZATION	I-1
	SYSTEM APPLICATION	I-1
	EQUIPMENT ENVIRONMENT	I-3
	PROGRAM ENVIRONMENT	I-3
II	DATA BASE MAINTENANCE EXECUTIVES AND PROGRAMS	
	INTRODUCTION	II-1
	GENERAL SYSTEM PROCEDURES	II-2
	RAMIS FILE DESCRIPTION - FACILITIES FILE	II-3
	RAMIS FILE DESCRIPTION - INSTRUCTOR FILE	II-6
	RAMIS FILE DESCRIPTION - COURSE FILE	II-10
	EXECUTIVE SEQUENCE DOCUMENTATION	II-16
	DATA BASE EXECUTIVE INDEX	II-23
	FACILITY FILE EXECUTIVE SEQUENCES	II-25
	INSTRUCTOR FILE EXECUTIVE SEQUENCES	II-60
	COURSE FILE EXECUTIVE SEQUENCES	II-117
	PGM5	II-168
	PGM7	II-177
	PGM8	II-182
	PGM9B	II-196
	PGM16	II-202
	PGM19	II-212

TABLE OF CONTENTS

SECTION		PAGE
	PGM20	II-229
	PGM22	II-235
	PROMPTX	II-243
III	ETE MODEL	
	MODEL DESCRIPTION	III-1
	EXECUTIVE SUBSYSTEM	III-6
	MODEL GENERATION SUBSYSTEM	III-15
	MODEL UPDATE	III-38
	SIMULATION SUBSYSTEM	III-60
	SIMULATION OUTPUT	III-62
IV	SYSTEM CAPABILITIES/REQUIREMENTS AND RESOURCES MODEL	
	MODEL DESCRIPTION	IV-1
	SCRR EXEC DESCRIPTION	IV-5
	RAMSCRR EXEC DESCRIPTION	IV-8
	SCRR INST DESCRIPTION	IV-8
	SCRR FACIL DESCRIPTION	IV-8
	SCRR FAVAIL DESCRIPTION	IV-9
	SCRR IAVAIL DESCRIPTION	IV-9
	SCRR CRSE DESCRIPTION	IV-10
	BLANK EXEC DESCRIPTION	IV-14
	BLANK PLI DESCRIPTION	IV-14
	CONCAT PLI DESCRIPTION	IV-14
	INTERFACE (IF1, IF2, IF3, AND IF4) EXECS OVERVIEW	IV-20

6

TABLE OF CONTENTS

SECTION	PAGE
IF1 EXEC DESCRIPTION	IV-22
IF1 FORTRAN DESCRIPTION	IV-22
IF2 EXEC DESCRIPTION	IV-27
INTERFAC FORTRAN DESCRIPTION	IV-27
SCRRSORT EXEC DESCRIPTION	IV-35
IF3 EXEC DESCRIPTION	IV-36
IF3 FORTRAN DESCRIPTION	IV-36
IF4 EXEC DESCRIPTION	IV-41
IF4 FORTRAN DESCRIPTION	IV-41
COMB EXEC DESCRIPTION	IV-45
SCRR1 EXEC DESCRIPTION	IV-46
SCRR RUN DESCRIPTION	IV-46
SCRRFORM FORTRAN DESCRIPTION	IV-49
ASSIGN EXEC DESCRIPTION	IV-65
ASSIGN1 FORTRAN DESCRIPTION	IV-65
ASSIGN2 FORTRAN DESCRIPTION	IV-66
MODAVAIL EXEC DESCRIPTION	IV-75
MODIFY FORTRAN DESCRIPTION	IV-75
REQMT EXEC DESCRIPTION	IV-81
REQMT RUNEXEC DESCRIPTION	IV-81
REQMT1 RUNEXEC DESCRIPTION	IV-81
REQMT2 RUNEXEC DESCRIPTION	IV-82
SCRR2 EXEC DESCRIPTION	IV-87
FIXCONV EXEC DESCRIPTION	IV-88
MAXCONV EXEC DESCRIPTION	IV-89

TABLE OF CONTENTS

SECTION		PAGE
V	TRAINING PROCESS FLOW MODEL	
	MODEL DESCRIPTION	V-1
	SUPPORT PROGRAMS	V-1
	DATA BASE SUPPORT PROGRAMS	V-3
	"WHAT IF" SUPPORT PROGRAMS	V-11
	COURSE MODIFICATIONS	V-11
	ACROSS THE BOARD MODIFICATIONS	V-15
	DISPLAY SUPPORT PROGRAM	V-71
	EXECUTIVE DISPLAY	V-71
	PROGRAM DISPLAY	V-71
	TPL LOGIC DESIGN	V-84
	PROGRAM LOGIC DESIGN	V-84
	TPF MODEL MODEL MAIN	V-84
	SUBROUTINE CRSLS	V-87
	SUBROUTINE PRERUN	V-87
	SUBROUTINE CRSEND	V-87
	SUBROUTINE QTREND	V-88
	SUBROUTINE YEARND	V-88
	SUBROUTINE CRTOUT	V-[illegible]
	SUBROUTINE NPRINT	V-88

8

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
III-1	ETE Model Operation Procedure Overview . . .	III-2
III-2	ETE Model Logic Overview	III-3
III-3	ETE Data File Summary	III-5
III-4	ETE2 and ETEPTR Executive Logic Flow	III-7
III-5	PROF Flowcharts	III-17
III-6	Input Card Specifications	III-37
III-8	MAD Flowcharts	III-40
III-9	System Generated GPSS Functions	III-61
IV-1	Overview of SCRR Executive Programs	IV-2
IV-2	SCRR EXEC Flowchart	IV-6
IV-3	RAMSCRR EXEC Flowchart	IV-11
IV-4	BLANK EXEC Flowcharts	IV-15
IV-5	Data Base INTERFACE EXEC Flowchart	IV-21
IV-6	IF1 Flowcharts	IV-23
IV-7	INTERFAC Flowcharts	IV-29
IV-8	IF3 Flowcharts	IV-37
IV-9	IF4 Flowcharts	IV-42
IV-10	SCRR EXEC Flowchart	IV-50
IV-11	ASSIGN EXEC Flowchart	IV-67
IV-12	MODAVAIL EXEC Flowchart	IV-76
IV-13	REQMNT EXEC Flowchart	IV-83
V-1	TPF Model Simplified System Flow	V-2
V-2	TPF Model Data Base Interface	V-4

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
V-3	TPF Input Formats	V-6
V-4	Flow Executive TPF COPY	V-7
V-5	TPF Model "WHAT IF" and Output Display . . .	V-12
V-6	Flow Executive TPFMOD	V-17
V-7	Autoflow Program TPFMOD	V-19
V-8	Autoflow Subroutine DECODE	V-29
V-9	Autoflow Subroutine CONVRT	V-42
V-10	Autoflow Subroutine MSG	V-44
V-11	Autoflow Subroutine SHOW	V-46
V-12	Flow Executive TPF	V-50
V-13	Autoflow Program PARAM	V-51
V-14	Autoflow Program DISPLAY	V-76
V-15	Course Scheduler Matrix	V-89
V-16	Sample TPF Outputs	V-90
V-17	Autoflow Program KEND	V-91

LIST OF TABLES

TABLE NO.	TITLE	PAGE
III-1	ETE Programs and Executives	III-4
III-2	ETE Executive Listings	III-9
III-3	PROF Program Listings	III-33
III-4	Sample Edit Commands	III-39
III-5	MAD Program Listings	III-57
III-6	DUMMY GPSS Listing	III-65
IV-1	SCRR Executive Programs Description	IV-3
IV-2	SCRR Data File Summary	IV-4
IV-3	SCRR Program Listing	IV-7
IV-4	RAMSCRR Program Listings	IV-12
IV-5	Blank Program Listings	IV-18
IV-6	IF1 Program Listings	IV-25
IV-7	IF2 Program Listings	IV-33
IV-8	SCRRSORT Program Listing	IV-35
IV-9	IF3 Program Listings	IV-39
IV-10	IF4 Program Listings	IV-43
IV-11	COMB Program Listing	IV-45
IV-12	SCRR1 Program Listings	IV-51
IV-13	ASSIGN Program Listings	IV-73
IV-14	MODAVAIL Program Listings	IV-79
IV-15	REQMT Program Listings	IV-84
IV-16	SCRR2 Program Listings	IV-87
IV-17	FIXCONV Program Listing	IV-88
IV-18	MAXCONV Program Listing	IV-89

LIST OF TABLES

TABLE NO.	TITLE	PAGE
V-1	RAMIS Dump Format (Master or Scratch)	V-5
V-2	TPFCOPY DECODE.	V-9
V-3	COPY FORTRAN	V-10
V-4	Listing Executive TPFMOD	V-57
V-5	Listing Program TPFMOD	V-58
V-6	Listing Subroutine DECODE	V-60
V-7	Listing Subroutine CONVRT	V64
V-8	Listing Subroutine MSG	V-65
V-9	Listing Subroutine SHOW	V-66
V-10	Subroutine COMND Listing	V-67
V-11	Listing Executive TPF	V-68
V-12	Listing Program PARAM	V-69
V-13	Listing Display Executives	V-73
V-14	Listing Program DISPLAY	V-74
V-15	Listing Program KEND	V-148

SECTION I

INTRODUCTION

PURPOSE

This Program Maintenance Manual is intended to provide detailed information on the current level of the three Design of Training Systems (DOTS) models and the DOTS data base. The level of information presented should be sufficient to provide a base for programmers intending to install or modify the DOTS System.

ORGANIZATION

This manual is organized around the four major sections of the DOTS System: the DOTS data base, the Educational Technology Evaluation (ETE) model, the System Capabilities Requirements and Resources (SCRR) model and the Training Process Flow (TPF) model. The three models function independently and the data base (although it serves both the SCRR and TPF models) can also be considered as an independent subsystem. Therefore, each of the models and the data base is discussed in a separate section of the manual.

Within each section the subsystem being discussed is subdivided into its major components. Each section contains a subsystem description and macro level flow, detailed flows and program listings of the subsystem components. For FORTRAN and PL1 programs the detailed flow charts were generated using the Autoflow II* System.

SYSTEM APPLICATION

The DOTS system consists of three models supported by a data base. Each model is of a different type and addresses a different facet in the development and management of training courses and systems. Figure I-1 is functional level diagram of the DOTS system showing the interaction of the models and the data base.

The DOTS system is an interactive system. Each of the three models and the data base can be updated and executed via the terminal equipment in the Command and Control Center (CCC). A brief description of the models is presented here. Detailed information on the models is contained in the body of this report and in Volume II of TAEG Report No. 12-2, "Design of Training Systems Phase II Report."

a) System Capabilities Requirements and Resources (SCRR) model. A linear programming model designed to determine the maximum student throughput for a training complex, based on an optimal mix of course convenings and available resources.

*Autoflow II is a product of Applied Data Research, Inc.

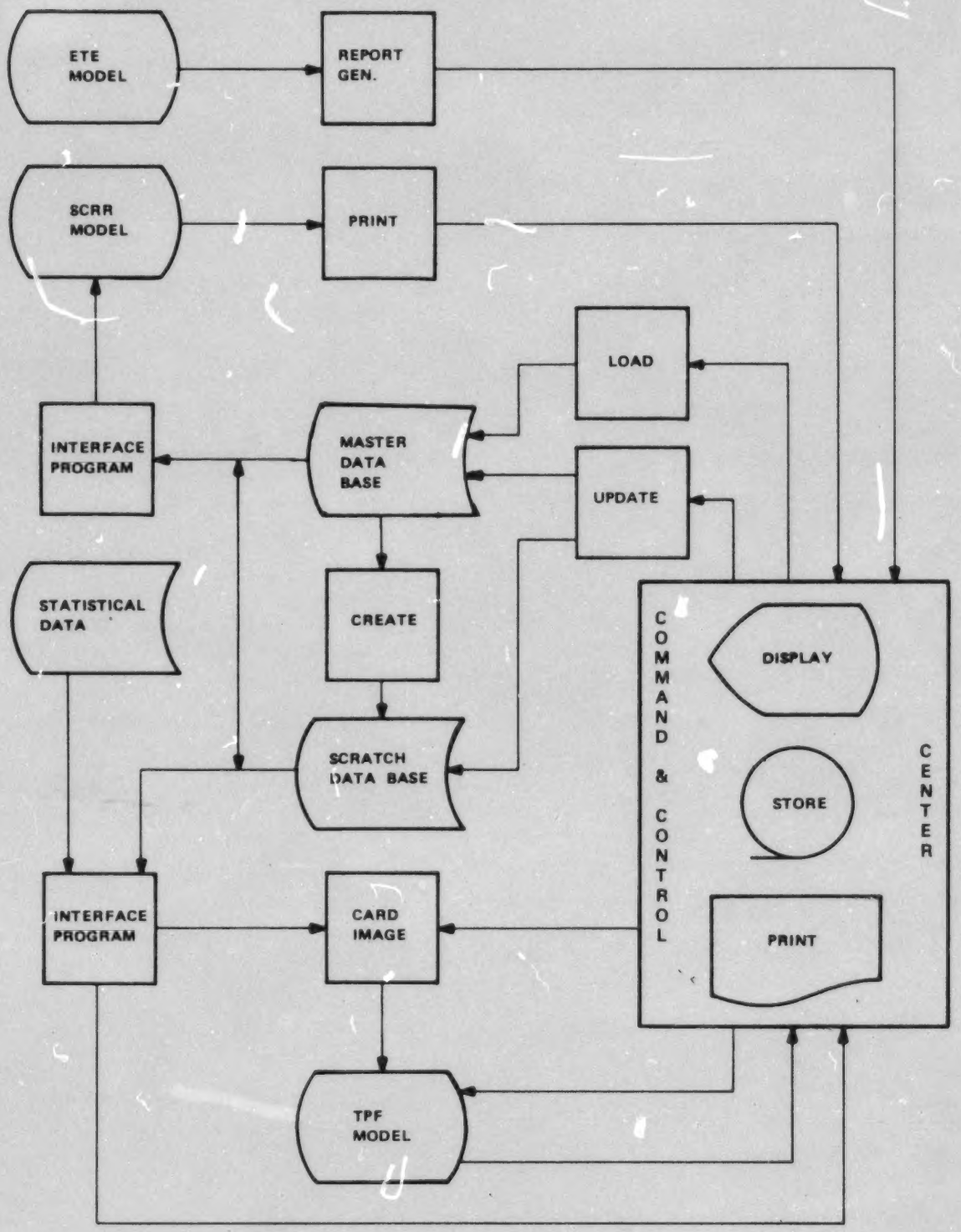


FIGURE I-1 DOTS SYSTEM DIAGRAM

b) Training Process Flow (TPF) model. The TPF model is an aggregate flow model designed to project the effect, on training system output, of changes in demand, scheduling, course capacity, and utilization.

c) Educational Technology Evaluation (ETE) model. A generalized entity flow simulation designed to project average time to complete and resource utilization for Individualized Learning Systems.

d) The DOTS data base provides a common, direct access data source for both the SCRR and TPF models. The DOTS data base was constructed and is accessed via the RAMIS® system, a data base language developed by Mathematica, Inc.

EQUIPMENT ENVIRONMENT

All DOTS programs are designed to run, in an interactive mode, on a remote time-shared computer. Time sharing services and support were obtained on a lease basis from National CSS, Inc., Norwalk, Connecticut. The National CSS system is described further under Program Environment.

The only hardware required by the user is a standard remote terminal such as an IBM 2741, a Hazeltine 2000[1], or a Teleterm 1030[2]. The terminal is connected to a dial-up telephone line via a modem. Further information on the different types of terminal equipment which can be employed in conjunction with the time-sharing support should be obtained from National CSS, Inc.

As currently configured, the DOTS system uses thirty-six cylinders of on-line disk storage and runs in a 320K byte partition of main memory.

PROGRAM ENVIRONMENT

In addition to the National CSS time-sharing software, the DOTS system employs several other program packages, all of which run under control of the VP/CSS[3] system. The program packages are:

Mathematical Programming System III: MPS III[4]
Rapid Access Management Information System: RAMIS[5]
Norden General Purpose Simulation System V: NGPSSV[6]

---

1. Trademark of Hazeltine Corp.
2. Trademark of Computer Devices, Inc.
3. Trademark of National CSS, Inc.
4. Trademark of Management Science Systems, Inc.
5. Trademark of Mathematica, Inc.
6. Trademark of Norden Division of United Aircraft Corporation

Information pertinent to DOTS operations can be found in the following publications:

A Guide to the National CSS Computer System for Application Product Users (NCSS Form 991)

VP/CSS Edit Command (NCSS Form 108)

Report Generation VP/CSS RAMIS (NCSS Form 961)

Installer's Guide VP/CSS RAMIS (NCSS Form 963)

GPSS/Norden Simulation Language (Reprinted by NCSS)

MPS III User Manual (Reprinted by NCSS)

Basic system operations such as login, Initial Program Load, system interrupts, etc. are covered in NCSS Form 991 (see above) sections 1 and 7. Consequently, they will not be reproduced in this Program Maintenance Manual.

SECTION II

DATA BASE MAINTENANCE EXECUTIVES AND PROGRAMS

INTRODUCTION

Documentation relative to the software elements developed for creation and maintenance of the three DOTS data bases is included within this section. First, are the RAMIS data management system file descriptions, followed by Executive Sequence details. Program documentation completes the section.

The NCSS Executives contained herein are termed "Executive Sequences", since most are several individual VP/CSS Executives in sequence. Due to the interactive attributes of the DOTS system, Executives are documented herein by reference to the Root Executive, or that to be invoked by the User at the terminal.

Program documentation contains a functional description of the task to be performed, input/output data formats (where applicable), and machine-drawn flowcharts and program source code listings.

Interaction of the User with the data base maintenance Executive Sequences and programs is covered in detail in the DOTS USER'S MANUAL, and is not, in general, reproduced in any detail within this section.

GENERAL SYSTEM PROCEDURES

The implications associated with the following RAMIS file descriptions and VP/CSS Executive Sequences presupposes a rather detailed knowledge on the part of the User with basic services provided by the National CSS (NCSS) time-sharing vendor. These services are formally documented by NCSS and are not reproduced within this manual. To assist those who are unfamiliar with the NCSS documentation, the following references are mentioned:

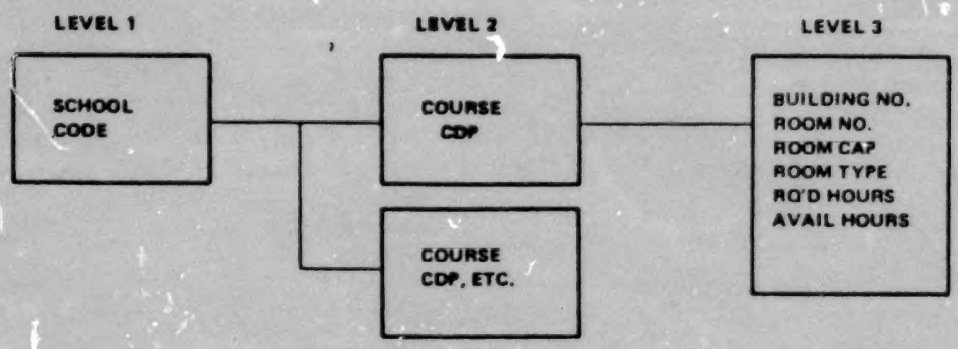
System Function	NCSS Reference
Typing Conventions, Login, Logout	A Guide to the National CSS Computer System for Application Product Users (NCSS Form 991) Section 1
System Interruptions	A Guide to the National CSS Computer System for Application Product Users (NCSS Form 991) Section 7
CSS Commands Summary	A Guide to the National CSS Computer System for Application Product Users (NCSS Form 991) Appendix B
CSS File Creation and Maintenance	A Guide to the National CSS Computer System for Application Product Users (NCSS Form 991) Section 3
	VP/CSS Edit Command (NCSS Form 108)
Data Base Report Generation	Report Generation VP/CSS RAMIS (NCSS Form 961)
Data Base Maintenance - Scan Mode	Installer's Guide UP/CSS RAMIS (NCSS Form 963) pages 3-37 - 3-84
RAMIS System RUN PROCEDURES	RAMIS Users Guide to the RUN EXECUTIVE (A Publication of MATHEMATICA, Inc., Princeton Station Office Park, P. O. Box 2392, Princeton, N. J. 08540)
VP/CSS EXECUTIVE Language	VP/CSS EXECUTIVE Language (EXEC) (NCSS Form 109-5)

RAMIS FILE DESCRIPTION

DATA BASE (RAMIS NAME): Facilities Master (FACIL)

Facilities Scratch (MODFACIL)

The tree-structure of the DOTS Facilities data bases (Master and Scratch) is as illustrated in the following diagram:



DOTS FACILITIES FILE – HIERARCHTIC STRUCTURE

The RAMIS Descriptions supporting the above structure for the Master and Scratch versions of the Facilities data bases are:

Data Base	Description Filename	Description Filetype
Facilities Master	FACIL	DESC
Facilities Scratch	MODFACIL	DESC

Listings of these two descriptions follow:

Listing of Filename = FACIL, Filetype = DESC

```
ERASE*
FILE FACIL
END
INPUT
FILE RAMASTER
1=FACIL,1,S,12,SCH,SCH,A,4,$
2=2,S,5,CDP,CDP,A,4,$
2=3,S,5,BLDG,BLDG,A,5,$
5=RM,RM,A,3,$
5=RMCAP,RMCAP,I,3,$
5=RMTP,RMTP,A,1,$
5=REQHRS,REQHRS,F,6.1,$
5=AVHRS,AVHRS,I,5,$
END
```

Listing of Filename = MODFACIL, Filetype = DESC

```
ERASE*
FILE MODFACIL
END
INPUT
FILE RAMASTER
1=MODFACIL,1,S,12,SCH,SCH,A,4,$
2=2,S,5,CDP,CDP,A,4,$
2=3,S,5,BLDG,BLDG,A,5,$
5=RM,RM,A,3,$
5=RMCAP,RMCAP,I,3,$
5=RMTP,RMTP,A,1,$
5=REQHRS,REQHRS,F,6.1,$
5=AVHRS,AVHRS,I,5,$
END
```

A sample output from the RAMIS procedure RAMINDEX is shown below for the Facilities Master data base (FACIL). (A RAMINDEX run for the Scratch version would have been identical.)

DESCRIPTION FOR RAMIS FILE NAMED FACIL Ø9/18/75

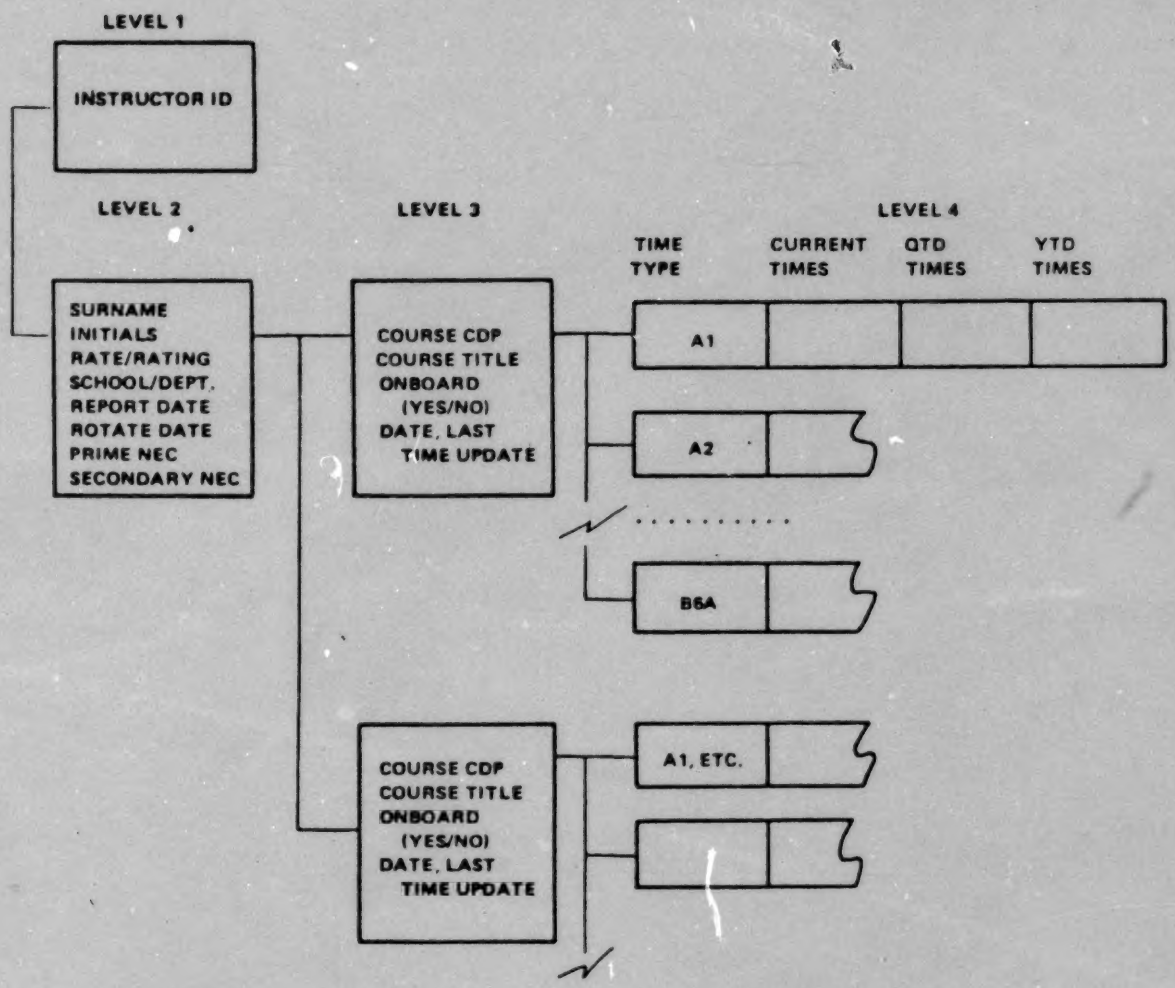
LIST	FIELD NAME	ALIAS	TYPE	LENGTH	LEVEL	SEGMENT	SORT FLAG
1	SCH	SCH	A	4	1	12	S
2	CDP	CDP	A	4	2	5	S
3	BLDG	BLDG	A	5	3	5	S
4	RM	RM	A	3	3	5	S
5	RMCAP	RMCAP	I	3	3	5	S
6	RMTP	RMTP	A	1	3	5	S
7	REQHRS	REQHRS	F	6.1	3	5	S
8	AVHRS	AVHRS	I	5	3	5	S

RAMIS FILE DESCRIPTION

DATA BASE (RAMIS NAME): Instructor Master (INST)

Instructor Scratch (MODINST)

The tree-structure of the DOTS Instructor data bases (Master and Scratch) is as illustrated in the following diagram:



DOTS INSTRUCTOR FILE – HIEARCHTIC STRUCTURE

The RAMIS Descriptions supporting the above structure for the Master and Scratch versions of the Instructor data bases are:

Data Base	Description Filename	Description Filetype
Instructor Master	INST	DESC
Instructor Scratch	MODINST	DESC

Listings of these two descriptions follow:

Listing of Filename = INST, Filetype = DESC

```
ERASE*
FILE INST
END
INPUT
FILE RAMASTER
1=INST,1,S,50,ID,ID,A,9,$
2=2,,1,NME,NME,A,16,$
5=IN,IN,A,2,$
5=RATE,RATE,A,5,$
5=SCH,SCH,A,4,$
5=REP,REP,I,4,$
5=ROT,ROT,I,4,$
5=PNEC,PNEC,A,4,$
5=SNEC,SNEC,A,4,$
2=3,S,5,CDP,CDP,A,4,$
5=TTL,TTL,A,16,$
5=ONB,ONB,A,1,$
5=LDTE,LDTE,I,6,$
2=4,,14,TT,TT,A,3,$
5=CW,CW,F,4.1,$
5=QTD,QTD,F,5.1,,,,1,$
5=YTD,YTD,F,6.1,,,,1,$
END
```

Listing of Filename = MODINST, Filetype = DESC

```
ERASE*
FILE MODINST
END
INPUT
FILE RAMASTER
1=MODINST,1,S,50,ID,ID,A,9,$
2=2,,1,NME,NME,A,16,$
5=IN,IN,A,2,$
5=RATE,RATE,A,5,$
5=SCH,SCH,A,4,$
5=REP,REP,I,4,$
5=ROT,ROT,I,4,$
5=PNEC,PNEC,A,4,$
5=SNEC,SNEC,A,4,$
2=3,S,5,CDP,CDP,A,4,$
5=TTL,TTL,A,16,$
5=ONB,ONB,A,1,$
5=LDTE,LDTE,I,6,$
2=4,,14,TT,TT,A,3,$
5=CW,CW,F,4.1,$
5=QTD,QTD,F,5.1,,,,1,$
5=YTD,YTD,F,6.1,,,,1,$
END
```

A sample output from the RAMIS procedure RAMINDEX is shown below for the Instructor Master data base (INST). (A RAMINDEX run for the Scratch version would have been identical.)

DESCRIPTION FOR RAMIS FILE NAMED INST Ø9/18/75

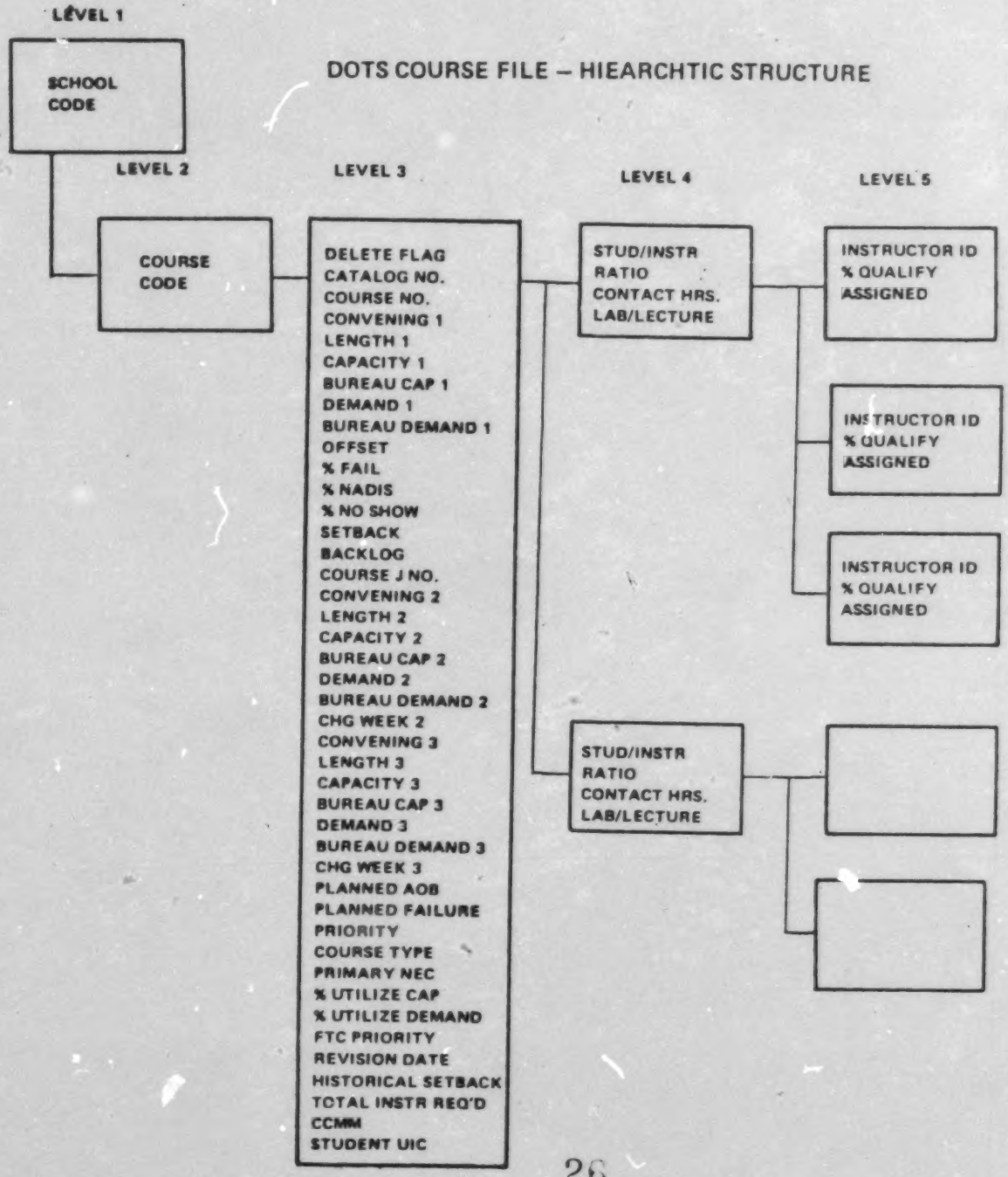
LIST	FIELD NAME	ALIAS	TYPE	LENGTH	LEVEL	SEGMENT	SORT FLAG
1	ID	ID	A	9	1	5Ø	S
2	NME	NME	A	16	2	1	
3	IN	IN	A	2	2	1	
4	RATE	RATE	A	5	2	1	
5	SCH	SCH	A	4	2	1	
6	REP	REP	I	4	2	1	
7	ROT	ROT	I	4	2	1	
8	PNEC	PNEC	A	4	2	1	
9	SNEC	SNEC	A	4	2	1	
1Ø	CDP	CDP	A	4	3	5	S
11	TTL	TTL	A	16	3	5	S
12	ONB	ONB	A	1	3	5	S
13	LDTE	LDTE	I	6	3	5	S
14	TT	TT	A	3	4	14	
15	CW	CW	F	4.1	4	14	
16	QTD	QTD	F	5.1	4	14	
17	YTD	YTD	F	6.1	4	14	

RAMIS FILE DESCRIPTION

DATA BASE (RAMIS NAME): Course Master (COURSE)

Course Scratch (MODCOURSE)

The tree-structure of the DOTS Course data bases (Master and Scratch) is as illustrated in the following diagram:



The RAMIS Descriptions supporting the above structure for the Master and Scratch versions of the Course data bases are:

Data Base	Description Filename	Description Filetype
Course Master	COURSE	DESC
Course Scratch	MODCOURSE	DESC

Listing of Filename = COURSE, Filetype = DESC

```
ERASE*
FILE COURSE
END
INPUT
FILE RAMASTER
1=COURSE,1,S,12,SCH,SCH,A,4,$
2=2,S,5,CDP,CDP,A,4,$
2=3,,1,DEL,DEL,A,1,$
5=CTLG,CTLG,A,10,$
5=CRS,CRS,A,16,$
5=CONV1,CONV1,I,3,$
5=LEN1,LEN1,F,4.1,$
5=CAP1,CAP1,I,3,$
5=BCAP1,BCAP1,I,3,$
5=DMD1,DMD1,I,5,$
5=BDMD1,BDMD1,I,5,$
5=OFST,OFST,I,2,$
5=PFAIL,PFAIL,F,5.3,$
5=NADIS,NADIS,F,5.3,$
5=NOSHOW,NOSHOW,F,5.3,$
5=SETB,SETB,F,5.3,$
5=BKLG,BKLG,I,4,$
5=JNO,JNO,A,4,$
5=CONV2,CONV2,I,3,$
5=LEN2,LEN2,F,4.1,$
5=CAP2,CAP2,I,3,$
5=BCAP2,BCAP2,I,3,$
5=DMD2,DMD2,I,5,$
5=BDMD2,BDMD2,I,5,$
5=CHWK2,CHWK2,I,3,$
5=CONV3,CONV3,I,3,$
5=LEN3,LEN3,F,4.1,$
5=CAP3,CAP3,I,3,$
5=BCAP3,BCAP3,I,3,$
5=DMD3,DMD3,I,5,$
5=BDMD3,BDMD3,I,5,$
5=CHWK3,CHWK3,I,3,$
5=PAOB,PAOB,I,4,$
5=AFAIL,AFAIL,F,5.3,$
5=PRTY,PRTY,I,2,$
5=CTYPE,CTYPE,A,2,$
5=PNEC,PNEC,A,4,$
5=UCAP,UCAP,I,3,$
5=UDEM,UDEM,I,3,$
5=FPRTY,FPRTY,I,2,$
5=RDTE,RDTE,I,6,$
5=HSETB,HSETB,F,5.3,$
5=TINSTR,TINSTR,F,4.1,$
5=SUIC,SUIC,A,5,$
5=CCMM,CCMM,A,10,$
2=4,,2,SIR,SIR,F,4.1,$
5=CHRS,CHRS,F,5.1,$
5=CTYP,CTYP,A,1,$
2=5,,3,ID,ID,A,9,$
5=QUAL,QUAL,A,3,$
```

28

Listing of Filename = MODCOURSE, Filetype = DESC

```
ERASE*
FILE MODCOURSE
END
INPUT
FILE RAMASTER
1=MODCOURSE,1,S,12,SCH,SCH,A,4,$
2=2,S,5,CDP,CDP,A,4,$
2=3,,1,DEL,DEL,A,1,$
5=CTLG,CTLG,A,10,$
5=CRS,CRS,A,16,$
5=CONV1,CONV1,I,3,$
5=LEN1,LEN1,F,4.1,$
5=CAP1,CAP1,I,3,$
5=BCAP1,BCAP1,I,3,$
5=DMD1,DMD1,I,5,$
5=BDMD1,BDMD1,I,5,$
5=OFST,OFST,I,2,$
5=PFAIL,PFAIL,F,5.3,$
5=NADIS,NADIS,F,5.3,$
5=NOSHOW,NOSHOW,F,5.3,$
5=SETB,SETB,F,5.3,$
5=BKLG,BKLG,I,4,$
5=JNO,JNO,A,4,$
5=CONV2,CONV2,I,3,$
5=LEN2,LEN2,F,4.1,$
5=CAP2,CAP2,I,3,$
5=BCAP2,BCAP2,I,3,$
5=DMD2,DMD2,I,5,$
5=BDMD2,BDMD2,I,5,$
5=CHWK2,CHWK2,I,3,$
5=CONV3,CONV3,I,3,$
5=LEN3,LEN3,F,4.1,$
```

Listing of Filename = MODCOURSE, Filetype = DESC (Continued)

```
5=CAP3,CAP3,I,3,$
5=BCAP3,BCAP3,I,3,$
5=DMD3,DMD3,I,5,$
5=BDMD3,BDMD3,I,5,$
5=CHWK3,CHWK3,I,3,$
5=PAOB,PAOB,I,4,$
5=AFAIL,AFAIL,F,5.3,$
5=PRTY,PRTY,I,2,$
5=CTYPE,CTYPE,A,2,$
5=PNEC,PNEC,A,4,$
5=UCAP,UCAP,I,3,$
5=UDEM,UDEM,I,3,$
5=FPRTY,FPRTY,I,2,$
5=RDTE,RDTE,I,6,$
5=HSETB,HSETB,F,5.3,$
5=TINSTR,TINSTR,F,4.1,$
5=SUIC,SUIC,A,5,$
5=CCMM,CCMM,A,10,$
2=4,,2,SIR,SIR,F,4.1,$
5=CHRS,CHRS,F,5.1,$
5=CTYP,CTYP,A,1,$
2=5,,3,ID,ID,A,9,$
5=QUAL,QUAL,A,3,$
5=ASN,ASN,I,1,$
2=6,V,1,NME,NME,A,16,,ID,MODINST,,1 2,$
END
```

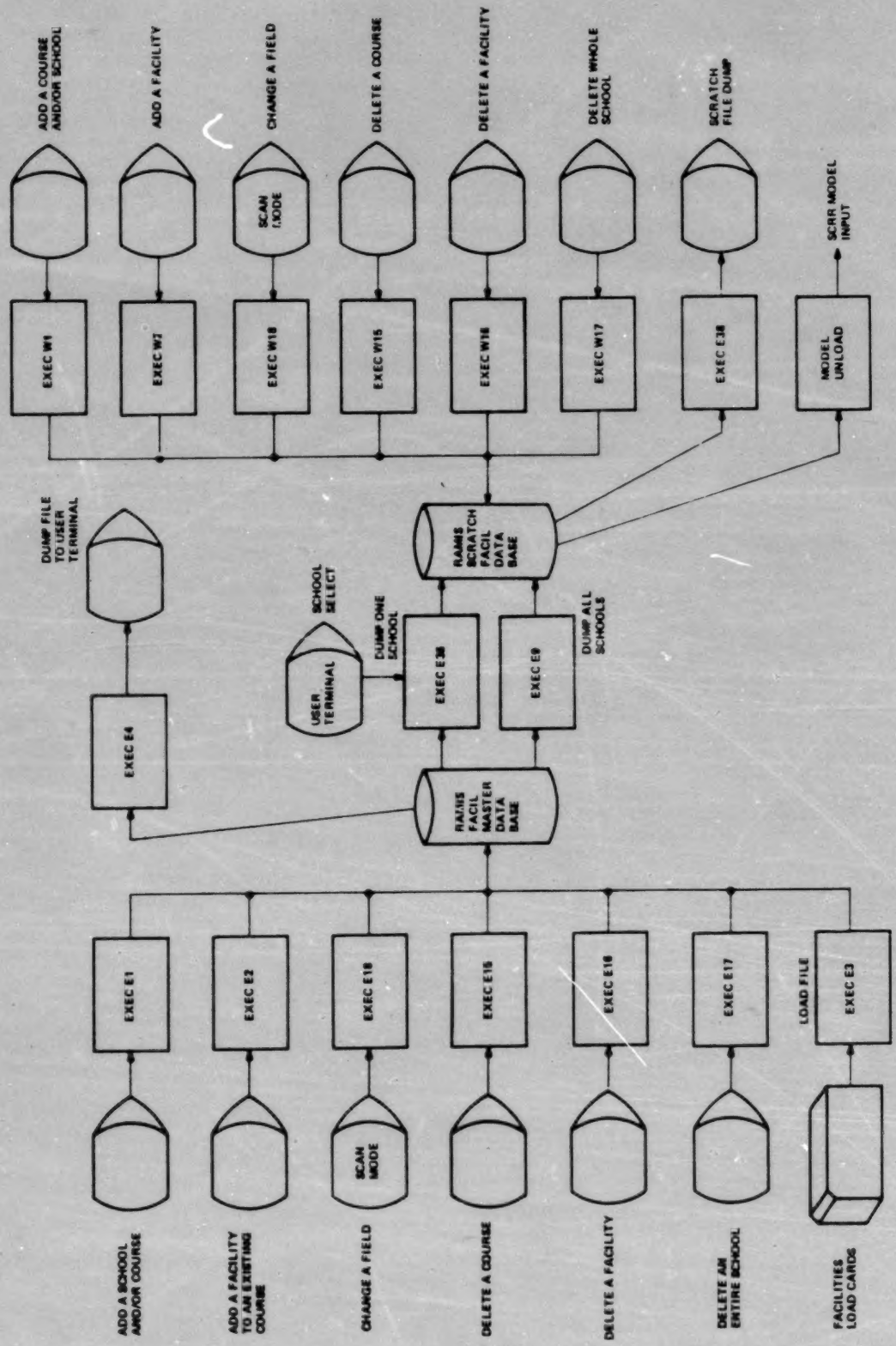
A sample output from the RAMIS procedure RAMINDEX is shown below for the Course Master data base (COURSE). (A RAMINDEX run for the Scratch version would have been identical.)

DESCRIPTION FOR RAMIS FILE NAMED COURSE Ø9/18/75

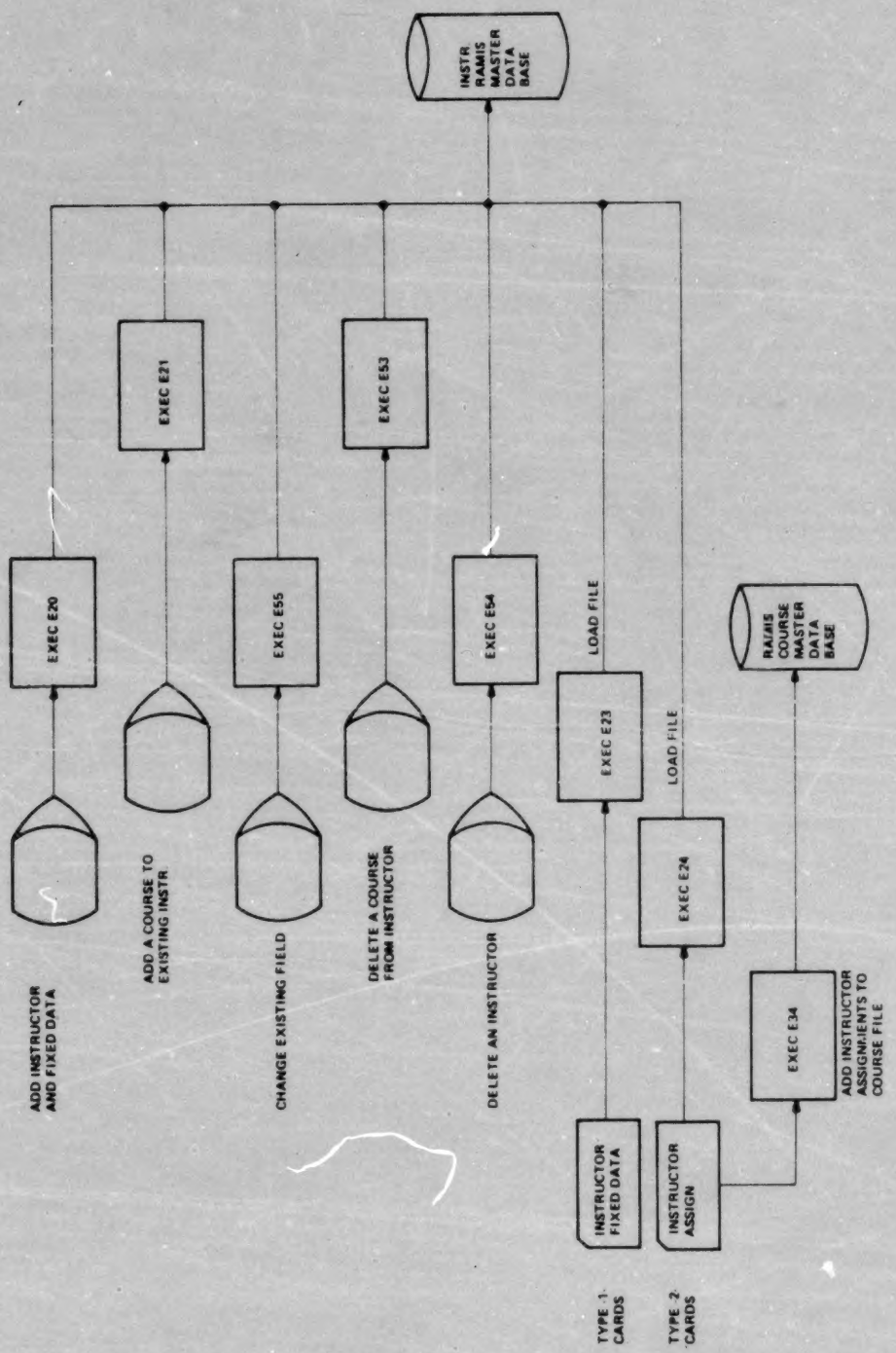
LIST	FIELD NAME	ALIAS	TYPE	LENGTH	LEVEL	SEGMENT	SORT FLAG
1	SCH	SCH	A	4	1	12	S
2	CDP	CDP	A	4	2	5	S
3	DEL	DEL	A	1	3	1	
4	CTLG	CTLG	A	1Ø	3	1	
5	CRS	CRS	A	16	3	1	
6	CONV1	CONV1	I	3	3	1	
7	LEN1	LEN1	F	4.1	3	1	
8	CAP1	CAP1	I	3	3	1	
9	BCAP1	BCAP1	I	3	3	1	
1Ø	DMD1	DMD1	I	5	3	1	
11	BDMD1	BDMD1	I	5	3	1	
12	OFST	OFST	I	2	3	1	
13	PFAIL	PFAIL	F	5.3	3	1	
14	NADIS	NADIS	F	5.3	3	1	
15	NOSHOW	NOSHOW	F	5.3	3	1	
16	SETB	SETB	F	5.3	3	1	
17	BKLG	BKLG	I	4	3	1	
18	JNO	JNO	A	4	3	1	
19	CONV2	CONV2	I	3	3	1	
2Ø	LEN2	LEN2	F	4.1	3	1	
21	CAP2	CAP2	I	3	3	1	
22	BCAP2	BCAP2	I	3	3	1	
23	DMD2	DMD2	I	5	3	1	
24	BDMD2	BDMD2	I	5	3	1	
25	CHWK2	CHWK2	I	3	3	1	
26	CONV3	CONV3	I	3	3	1	
27	LEN3	LEN3	F	4.1	3	1	
28	CAP3	CAP3	I	3	3	1	
29	BCAP3	BCAP3	I	3	3	1	
3Ø	DMD3	DMD3	I	5	3	1	
31	BDMD3	BDMD3	I	5	3	1	
32	CHWK3	CHWK3	I	3	3	1	
33	PAOB	PAOB	I	4	3	1	
34	AFAIL	AFAIL	F	5.3	3	1	
35	PRTY	PRTY	I	2	3	1	
36	CTYPE	CTYPE	A	2	3	1	
37	PNEC	PNEC	A	4	3	1	
38	UCAP	UCAP	I	3	3	1	
39	UDEM	UDEM	I	3	3	1	
4Ø	FPRTY	FPRTY	I	2	3	1	
41	RDTE	RDTE	I	6	3	1	
42	HSETB	HSETB	F	5.3	3	1	
43	TINSTR	TINSTR	F	4.1	3	1	
44	SUIC	SUIC	A	5	3	1	
45	CCMM	CCMM	A	1Ø	3	1	
46	SIR	SIR	F	4.1	4	2	
47	CHRS	CHRS	F	5.1	4	2	
48	CTYP	CTYP	A	1	4	2	
49	ID	ID	A	9	5	3	
5Ø	QUAL	QUAL	A	3	5	3	
51	ASN	ASN	I	1	5	3	

EXECUTIVE SEQUENCE DOCUMENTATION

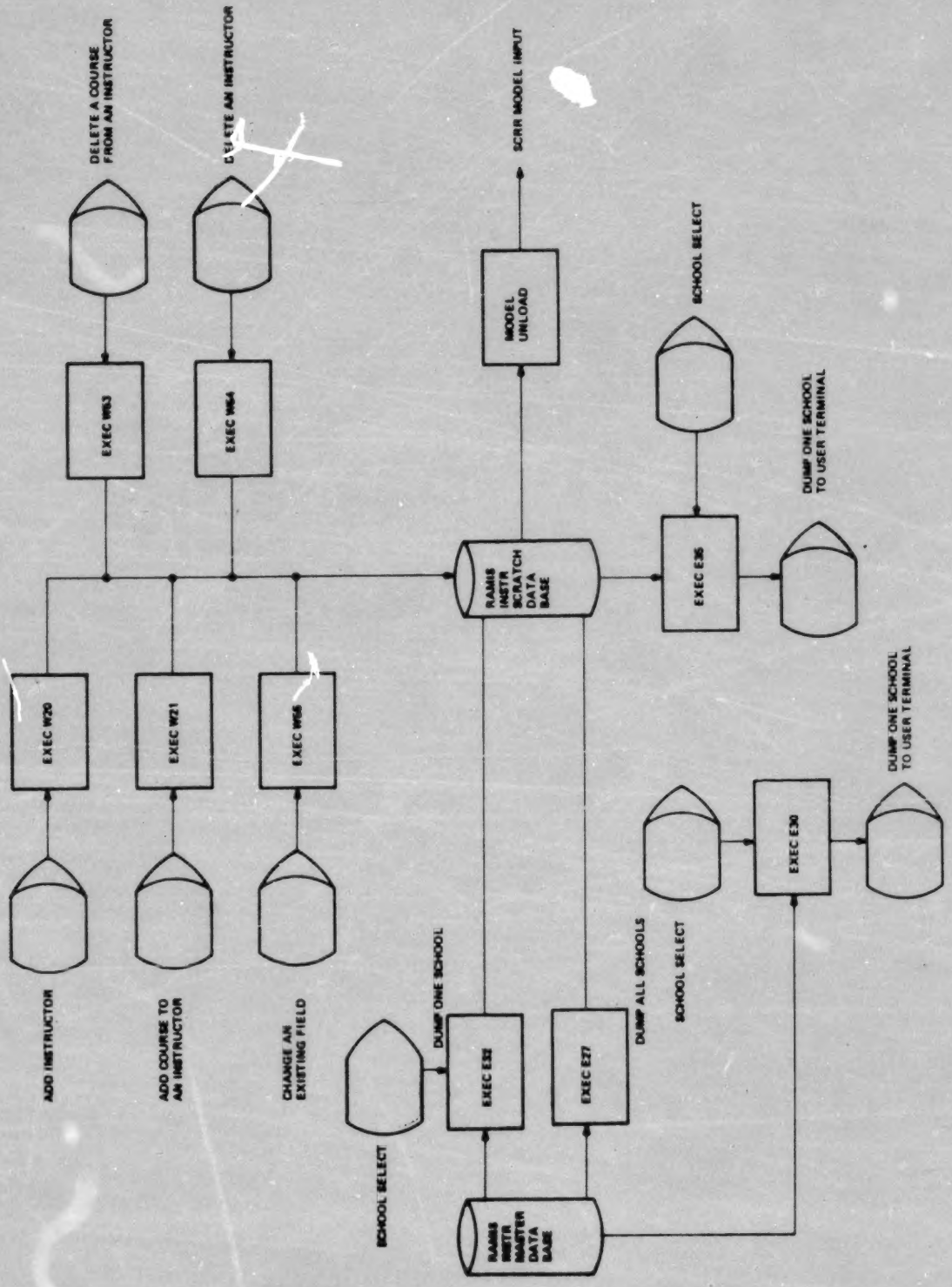
For each of the three DOTS data bases (Facilities, Instructor, and Course), diagrams follow which pictorially depict the action and utility of the Executive Sequences to be provided for data maintenance. These diagrams are the same that appear in the initial portion of the DOTS User's Manual. The order of appearance of the Executive Sequences within this section parallels that of the diagrams. An index titled "Data Base Executive Sequence and Program Index" follows the initial diagrams. This Index includes an alphabetic listing of the Root Executives and programs by name, and their page location within this section.



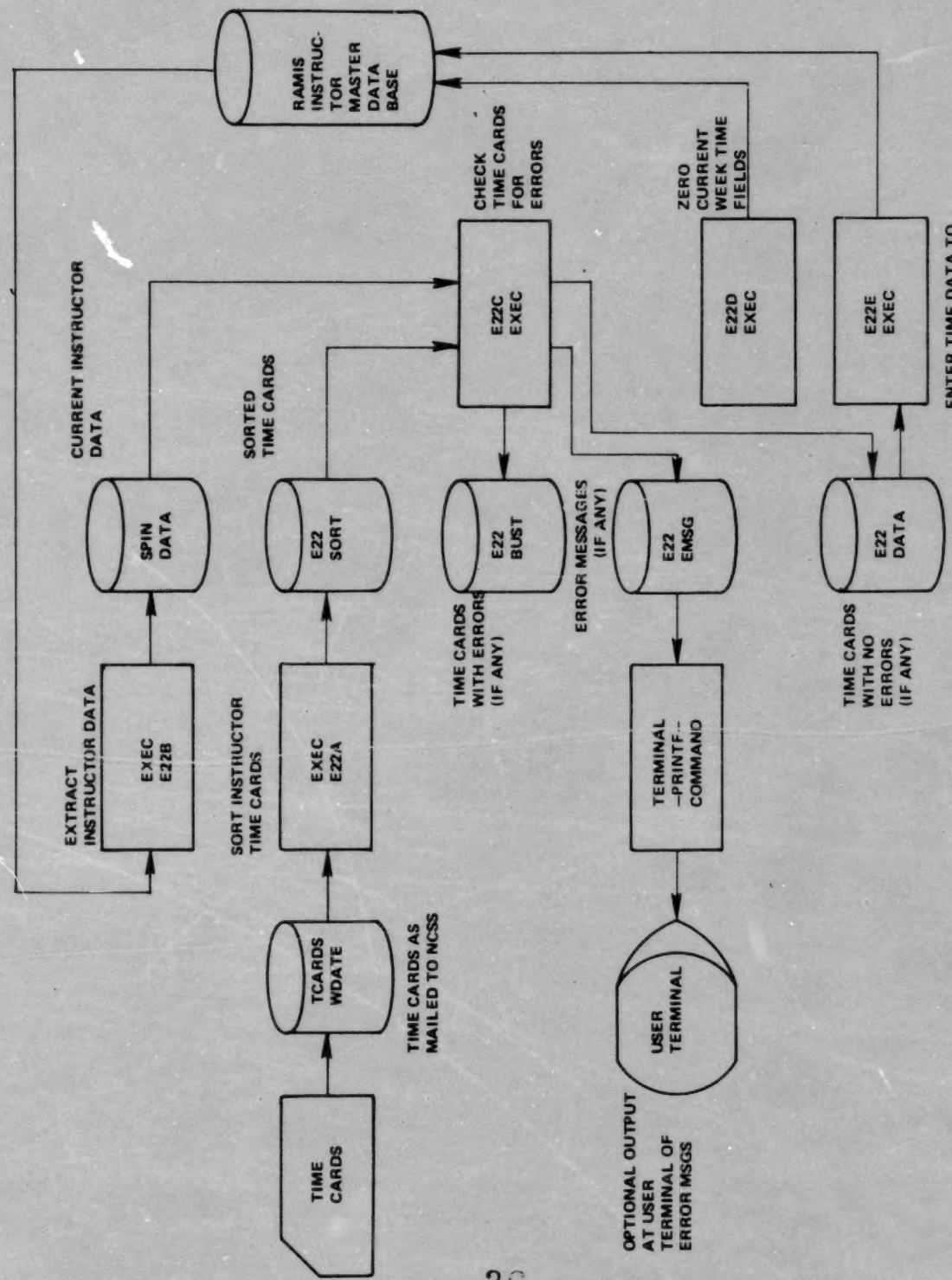
FACILITIES DATA BASE MAINTENANCE



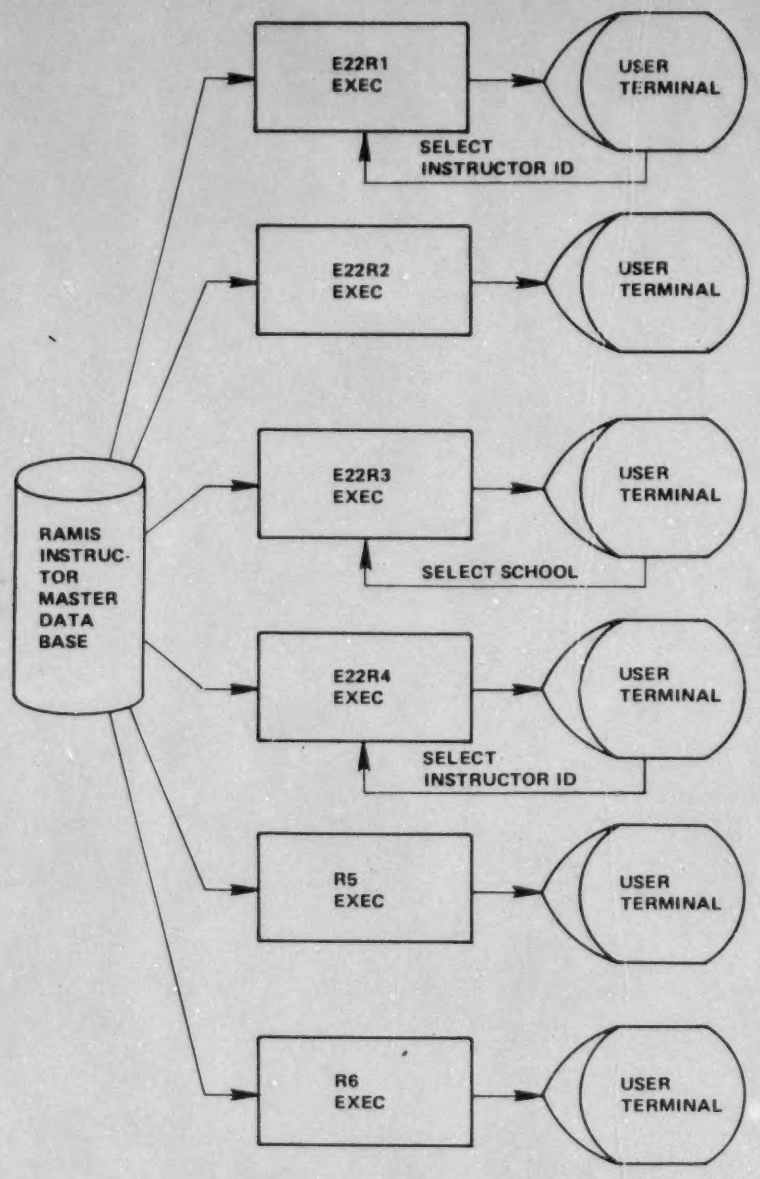
INSTRUCTOR DATA BASE MAINTENANCE



INSTRUCTOR DATA BASE MAINTENANCE



INSTRUCTOR DATA BASE WEEKLY TIME CARD PROCESSING



FOR EACH INSTRUCTOR SELECTED, LIST ID, NAME, AND CDP NUMBERS OF EACH COURSE ASSIGNED.

OUTPUTS WHOLE INSTRUCTOR FILE LISTING BY SCHOOL: INSTRUCTOR ID, NAME, ASSIGNED COURSE CDP NUMBERS.

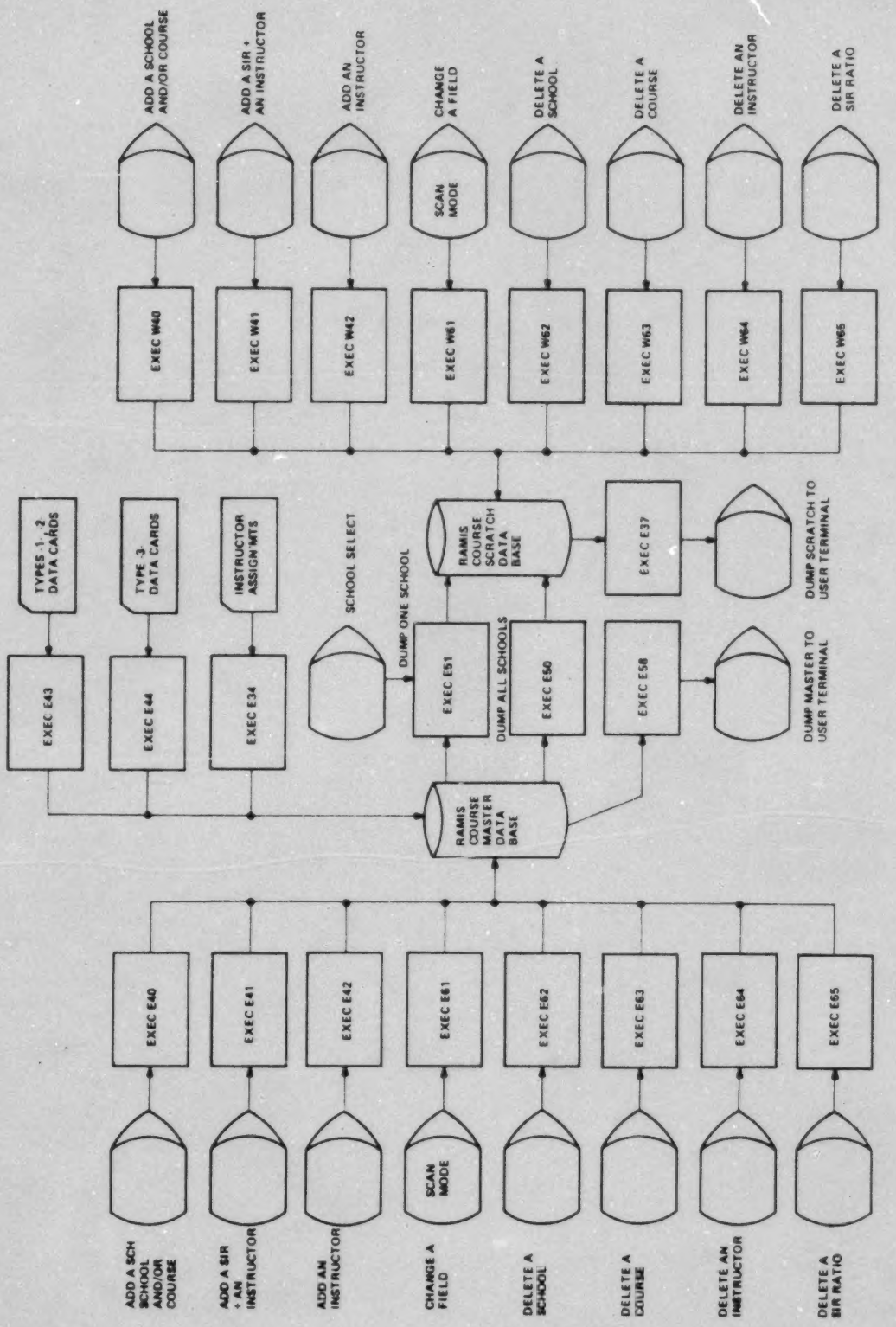
OUTPUTS, FOR THE SCHOOL SELECTED: INSTRUCTOR ID, NAME, AND SUM OF EACH INSTRUCTOR'S TIME EXPENDED DURING THE CURRENT WEEK.

FOR THE INSTRUCTOR SELECTED, LISTED ARE NAMES, ID, AND EACH OF THE COURSES' TIME TYPE, AND TIMES THAT HE HAS CHARGED DURING THE CURRENT WEEK.

FOR ALL SCHOOLS OF FTC, THE ROTATION DATES (MONTH/YEAR) AND THE NUMBERS OF INSTRUCTORS SCHEDULED TO ROTATE ON THOSE DATES ARE LISTED. ONLY THOSE ROTATIONS THAT WILL OCCUR IN THE NEXT 6 MONTHS ARE CONSIDERED.

ESSENTIALLY THE SAME USE AS REPORT R5, ABOVE, EXCEPT THAT THE NAME AND ID OF EACH INSTRUCTOR INVOLVED ARE INCLUDED.

INSTRUCTOR DATA BASE REPORTS



COURSE DATA BASE MAINTENANCE

DATA BASE EXECUTIVE INDEX

EXECUTIVE SEQUENCE	PAGE NO.	EXECUTIVE SEQUENCE	PAGE NO.	EXECUTIVE SEQUENCE	PAGE NO.
E1	II-25	E24	II-72	E64	II-130
E2	II-27	E27	II-88	E65	II-132
E3	II-38	E30	II-90	R5	II-113
E4	II-41	E32	II-85	R6	II-115
E9	II-45	E34	II-141	W1	II-47
E15	II-31	E35	II-93	W2	II-49
E16	II-33	E36	II-42	W15	II-52
E17	II-36	E37	II-150	W16	II-54
E18	II-30	E38	II-59	W17	II-57
E20	II-60	E40	II-117	W18	II-51
E21	II-62	E41	II-121	W20	II-75
E22A	II-96	E42	II-123	W21	II-79
E22B	II-97	E43	II-134	W40	II-151
E22C	II-98	E44	II-138	W41	II-155
E22D	II-100	E50	II-147	W42	II-157
E22E	II-101	E51	II-144	W53	II-77
E22F	II-104	E53	II-65	W54	II-82
E22G	II-105	E54	II-67	W55	II-84
E22R1	II-106	E55	II-64	W61	II-159
E22R2	II-108	E58	II-149	W62	II-160
E22R3	II-109	E61	II-125	W63	II-162
E22R4	II-111	E62	II-126	W64	II-164
E23	II-69	E63	II-128	W65	II-165

DATA BASE EXECUTIVE INDEX (Cont'd)

PROGRAMS	PAGE NO.
PGM5	II-168
PGM7	II-177
PGM8	II-182
PGM9B	II-196
PGM16	II-202
PGM19	II-212
PGM20	II-229
PGM22	II-235
PROMPTX . . .	II-243

ROOT EXECUTIVE: E1

DATA BASE: Facilities Master

FUNCTION: (1) Add a new Course to an existing School,/or/
(2) Add a new School and Course

The sequence described below forms a CLASS A Prompting Type Executive. The file PD1/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD1/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E1 Logic

The components of Executive Sequence E1 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	E1	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE1	RAM	

Listing of Filename = E1, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD1 RE1
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE1, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,5,3,3,1,6,5
KEY 1,2
INCLUDE 1
FILE FACIL
END
```

Listing of Filename = PD1, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER E1
1* FILE - FACILITIES
1* FUNCTION - ADD A NEW COURSE...
2SCH          11 4                    1              1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP          21 4                    1              5 4
3* ENTER COURSE CDP (CDP), A4
```

ROOT EXECUTIVE: E2

DATA BASE: Facilities Master

FUNCTION: Add a new Facility to an existing Course

The sequence described below forms a CLASS A Prompting Type Executive. The file PD2/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD2/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E2 Logic

The components of Executive Sequence E2 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	E2	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE2	RAM	

Listing of Filename = E2, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD2 RE2
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE2, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,5,3,3,1,6,5
KEY 1,2,3,4
INCLUDE 3
FILE FACIL
END
```

Listing of Filename = PD2, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER E2
1* FILE - FACILITIES
1* FUNCTION - ADD NEW FACILITY TO AN
1*            EXISTING COURSE...
2SCH        11 4                  1                          1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP        21 4                  1                          5 4
3* ENTER COURSE CDP (CDP), A4
2BLDG       31 5                  1                          9 5
3* ENTER BUILDING NUMBER (BLDG), A5
2RM         41 3                  1                         14 3
3* ENTER ROOM NUMBER (RM), A3
2RMCAP      52    000             220.0       200.0         17 3
3* ENTER ROOM CAPACITY (RMCAP), I3
2RMTP       61 1                  2                         20 1
3* ENTER ROOM TYPE (RMTP), A1
3*     1 = LECTURE, 2 = LAB, 3 = BOTH
2REQHRS     73    0000.0          220.0       3000.0      1 21 6
3* ENTER REQUIRED HOURS (REQHRS), F6.1
2AVHRS      82    0               220.0       3000.0        27 5
3* ENTER AVAILABLE HOURS (AVHRS), I5
```

ROOT EXECUTIVE: E18

DATA BASE: Facilities Master

FUNCTION: Change an existing Field

This single executive serves to force entry to the SCAN mode of the RAMIS data management system. After SCAN mode has been successfully entered, the User may direct change of existing field(s) interactively through his terminal.

Terminal entry requirements of the RAMIS SCAN mode are fully covered in the NCSS vendor's documentation.

The DOTS User's Manual lists the data field names and identification numbers as appropriate for the Facilities Master Data Base. This list will not be repeated in this document.

Listing of Executive E18 (SCAN Mode Entry)

```
&TYPE OFF
&STACK KT
&STACK SCAN
&STACK FILE FACIL
ATTACH RAMIS ALL BR
&STACK RT
RAMIS
DETACH RAMIS BR
&EXIT
```

ROOT EXECUTIVE: E15

DATA BASE: Facilities Master

FUNCTION: Delete a Course

The components of Executive Sequence E15 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	E15	EXEC
(2)	X15	RUNPROC

Listing of Filename = E15, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER E15
&PRI * FILE - FACILITIES
&PRI * FUNCTION - DELETE A COURSE
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &GOTO -OUT
&SAV1 = &1
-LP2 &PRI *
&PRI * ENTER COURSE CDP (CDP), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK2
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP2
-CK2 &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC X15 &SAV1 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = X15, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X15,$
END
CATALOG X15
DELETE
FORM 4,X4,4
SET 2
FILE FACIL
&1
END
END CATALOG
```

ROOT EXECUTIVE: E16

DATA BASE: Facilities Master

FUNCTION: Delete a Facility

The components of Executive Sequence E16 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	E16	EXEC
(2)	X16	RUNPROC

Listing of Filename = E16, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER E16
&PRI * FILE - FACILITIES
&PRI * FUNCTION - DELETE A FACILITY FROM A COURSE
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &GOTO -OUT
&SAV1 = &1
-LP2 &PRI *
&PRI * ENTER COURSE CDP (CDP), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK2
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP2
-CK2 &IF &1 EQ OUT &GOTO -OUT
&SAV2 = &1
-LP3 &PRI *
&PRI * ENTER BUILDING NO (BLDG), A5
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK3
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP3
-CK3 &IF &1 EQ OUT &GOTO -OUT
&SAV3 = &1
-LP4 &PRI *
&PRI * ENTER ROOM NO (RM),A3
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK4
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP4
-CK4 &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC X16 &SAV1 &SAV2 &SAV3 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = X16, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X16,$
END
CATALOG X16
DELETE
FORM 4,X4,4,X4,5,X3,3
SET 3
FILE FACIL
&1
END
END CATALOG
```

ROOT EXECUTIVE: E17

DATA BASE: Facilities Master

FUNCTION: Delete an entire School

The components of Executive Sequence E17 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	E17	EXEC
(2)	X17	RUNPROC

Listing of Filename = E17, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER E17
&PRI * FILE - FACILITIES
&PRI * FUNCTION - DELETE A SCHOOL
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC X17 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = X17, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X17,$
END
CATALOG X17
DELETE
FORM 4
SET 1
FILE FACIL
&1
END
END CATALOG
```

ROOT EXECUTIVE: E3

DATA BASE: Facilities Master

FUNCTION: Load the Facilities Master data base from card input

This Executive Sequence is used to create the Facilities Master data base in its entirety.

This sequence commences operation by causing the erase of any existing copy of the Facilities Master data base

User interface with this Executive Sequence is covered in the DOTS User's Manual.

The format of the card input data set used as an input to Executive Sequence E3 is illustrated in the diagram of this section entitled "Facilities Data Base Load Card Format."

Executive Sequence E3 Logic

The components of Executive Sequence E3 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype
(1)	E3	EXEC
(2)	RE3	RAM

Listing of Filename = E3, Filetype = EXEC

```
&TYPE OFF
STATE E3 DATA
&IF &INDEX0 NE 0 &GOTO -ERR1
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE3 RAM
DETACH RAMIS BR
ERASE E3 DATA
&EXIT
-ERR1 &PRINT NO FACILITIES LOAD CARDS FOUND - FUNCTION TERMINATIN
```

Listing of Filename = RE3, Filetype = RAM

```
ERASE
FILE FACIL
END
INPUT
READ E3 DATA
RECORD 80
SET 1
FORM 4,4,5,3,3,1,6,5
FILE FACIL
END
```

FACILITIES DATA BASE LOAD CARD FORMAT

FACILITIES FILE

IBM Federal Systems Division, Space Systems Center, Huntsville, Alabama

80 Column Card Punch Layout

NAME: LOAD CARD

JOB NO. | PROG. PACKAGE | WRITTEN AS | PUNCH AS | CARD PUNCH PRINT YES NO | VERIFY YES NO | DATE RUN OUT | SHEET OF

SCH	CDP	BLDG	RM	RM CAP	RPT	REQHRS	AVHRS
						XXXX.X	

955-1115-0

ROOT EXECUTIVE: E4

DATA BASE: Facilities Master

FUNCTION: Dump Facilities Master data base

This Executive sequence produces a formatted dump to the User's terminal of all of the records of the Facilities Master data base.

Executive Sequence E4 Logic

The components of Executive Sequence E4 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype
(1)	E4	EXEC
(2)	RE4	RAM

Listing of Filename = E4, Filetype = EXEC

```
&TYPE OFF
&STACK KT
ATTACH RAMIS ALL BRIEF
&STACK RT
EXEC RAMISIN RE4 RAM
DETACH RAMIS BRIEF
&EXIT
```

Listing of Filename = RE4, Filetype = RAM

```
TABLE
FILE FACIL
PRINT BLDG AND RM AND RMCAP AND RMTP AND REQHRS AND AVHRS
BY SCH BY CDP
ON SCH PAGE-BREAK
END
```

ROOT EXECUTIVE: E36

DATA BASE: Facilities Master
Facilities Scratch

FUNCTION: Recreate the Scratch data base using the records from one User-selected school in the Master data base

The first step in the recreation of the Scratch data base is the erase of the existing Scratch copy.

After being prompted, the User must supply the 4-character name of the school of interest.

The Master data base is then dumped into card-image format onto a temporary file, selecting only those records associated with the school chosen by the User. The temporary file is then used as input to recreate the Scratch data base in its entirety.

At the successful conclusion of this Executive Sequence the contents of the Master data base are unchanged. The new Scratch data base will contain only the records associated with the selected school.

Executive Sequence E36 Logic

The components of Executive Sequence E36 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype
(1)	E36	EXEC
(2)	E56	EXEC
(3)	X56	RUNPROC
(4)	E6	EXEC
(5)	RE6	RAM

Listing of Filename = E36, Filetype = EXEC

```
&TYPE OFF
EXEC E56
EXEC E6
```

Listing of Filename = E56, Filetype = EXEC

```
&TYPE OFF
&PRINT * FILE FACILITIES
&PRINT * FUNCTION - SELECT SCHOOL FOR MODEL INPUT
&PRINT * ENTER OUT FOR PREMATURE EXIT
-LP1 &PRI *
&PRINT * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &EXIT
FILEDEF RAMUTIL1 DSK FACIL DUMP LR 80 RE FB BL 800
&STACK KT
&STACK EXEC X56  &1
&STACK FIN
ATTACH RAMIS ALL BR
FILEDEF 6 MSG KILL
RAMIS
DETACH RAMIS BR
&STACK RT
FILEDEF RAMUTIL1 CLEAR
FILEDEF 6 CONO
&STACK KT
&STACK L /D=/
&STACK Q
EDIT MSG KILL
&PRI * &1 SCHOOL DUMPED TO DISK FILE
```

Listing of Filename = X56, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X56,$
END
CATALOG X56
DUMP
FILE FACIL
PRINT AVHRS AND SAVE
IF SCH IS &1
END
END CATALOG
```

Listing of Filename = E6, Filetype = EXEC

```
&TYPE OFF
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN RE6 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
&EXIT
```

Listing of Filename = RE6, Filetype = RAM

```
ERASE
FILE MODFACIL
END
INPUT
READ FACIL DUMP
RECORD 80
FORM 4,4,5,3,3,1,6,5
SET 1
SEARCH FORWARD
FILE MODFACIL
END
```

ROOT EXECUTIVE: E9

DATA BASE: Facilities Master
Facilities Scratch

FUNCTION: Recreate the Scratch data base using the records from all Schools in the Master data base

The first step in the recreation of the Scratch data base is the erase of the existing Scratch copy.

The Master is then dumped into card-image format onto a temporary dataset. This file is then used as input to recreate the Scratch data base in its entirety.

Executive Sequence E9 Logic

The components of Executive Sequence E9 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype
(1)	E9	EXEC
(2)	E5	EXEC
(3)	RE5	RAM
(4)	E6	EXEC
(5)	RE6	RAM

Listing of Filename = E9, Filetype = EXEC

```
&TYPE OFF
EXEC E5
EXEC E6
&EXIT
```

Listing of Filename = E5, Filetype = EXEC

```
&TYPE OFF
FILEDEF RAMUTIL1 DSK FACIL DUMP LR 80 RE FB BL 800
&STACK KT
ATTACH RAMIS ALL BRIEF
EXEC RAMISIN RE5 RAM
DETACH RAMIS BRIEF
FILEDEF RAMUTIL1 CLEAR
&STACK RT
&PRINT ALL SCHOOLS DUMPED FROM FACILITIES FILE
```

Listing of Filename = RE5, Filetype = RAM

```
DUMP
FILE FACIL
PRINT AVHRS AND SAVE
END
```

Listing of Filename = E6, Filetype = EXEC

```
&TYPE OFF
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN RE6 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
&EXIT
```

Listing of Filename = RE6, Filetype = RAM

```
ERASE
FILE MODFACIL
END
INPUT
READ FACIL DUMP
RECORD 80
FORM 4,4,5,3,3,1,6,5
SET 1
SEARCH FORWARD
FILE MODFACIL
END
```

ROOT EXECUTIVE: W1

DATA BASE: Facilities Scratch

FUNCTION: (1) Add a new Course to an existing School, /or/
(2) Add a new School and Course

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD1/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD1/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W1 Logic

The components of Executive Sequence W1 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	W1	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW1	RAM	

Listing of Filename = W1, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD1 RW1
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW1, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,5,3,3,1,6,5
KEY 1,2
INCLUDE 1
FILE MODFACIL
END
```

Listing of Filename = WPD1, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER W1
1* FILE - FACILITIES SCRATCH DATA BASE
1* FUNCTION - ADD A NEW COURSE...
2SCH        11 4                    1       1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP        21 4                    1       5 4
3* ENTER COURSE CDP (CDP), A4
```

ROOT EXECUTIVE: W2

DATA BASE: Facilities Scratch

FUNCTION: Add a new Facility to an existing Course

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD2/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD2/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W2 Logic

The components of Executive Sequence W2 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	W2	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW2	RAM	

Listing of Filename = W2, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD2 RW2
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW2, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,5,3,3,1,6,5
KEY 1,2,3,4
INCLUDE 3
FILE MODFACIL
END
```

Listing of Filename = WPD2, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER W2
1* FILE - FACILITIES SCRATCH DATA BASE
1* FUNCTION - ADD NEW FACILITY TO AN
1*             EXISTING COURSE...
2SCH        11 4                    1                          1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP        21 4                    1                          5 4
3* ENTER COURSE CDP (CDP), A4
2BLDG       31 5                    1                          9 5
3* ENTER BUILDING NUMBER (BLDG), A5
2RM         41 3                    1                         14 3
3* ENTER ROOM NUMBER (RM), A3
2RMCAP      52    000               220.0       200.0         17 3
3* ENTER ROOM CAPACITY (RMCAP), I3
2RMTP       61 1                    2                         20 1
3* ENTER ROOM TYPE (RMTP), A1
3*     1 = LECTURE, 2 = LAB, 3 = BOTH
2REQHRS     73    0000.0            220.0       3000.0      1 21 6
3* ENTER REQUIRED HOURS (REQHRS), F6.1
2AVHRS      82    0                 220.0       3000.0        27 5
3* ENTER AVAILABLE HOURS (AVHRS), I5
```

ROOT EXECUTIVE: W18

DATA BASE: Facilities Scratch

FUNCTION: Change an existing Field

This single executive serves to force entry to the SCAN mode of the RAMIS data management system. After SCAN mode has been successfully entered, the User may direct change of existing field(s) interactively through his terminal.

Terminal entry requirements of the RAMIS SCAN mode are fully covered in the NCSS vendor's documentation.

The DOTS User's Manual lists the data field names and identification numbers as appropriate for the Facilities Scratch data base. This list will not be repeated in this document.

Listing of Executive W18 (SCAN Mode Entry)

```
&TYPE OFF
&STACK KT
&STACK SCAN
&STACK FILE MODFACIL
ATTACH RAMIS ALL BR
&STACK RT
RAMIS
DETACH RAMIS BR
```

ROOT EXECUTIVE: W15

DATA BASE: Facilities Scratch

FUNCTION: Delete a Course

The components of Executive Sequence W15 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	W15	EXEC
(2)	WX15	RUNPROC

Listing of Filename = W15, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER W15
&PRI * FILE - MODFACIL
&PRI * FUNCTION - DELETE A COURSE
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &GOTO -OUT
&SAV1 = &1
-LP2 &PRI *
&PRI * ENTER COURSE CDP (CDP), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK2
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP2
-CK2 &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC WX15 &SAV1 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = WX15, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=WX15,$
END
CATALOG WX15
DELETE
FORM 4,X4,4
SET 2
FILE MODFACIL
&1
END
END CATALOG
```

ROOT EXECUTIVE: W16

DATA BASE: Facilities Scratch

FUNCTION: Delete a Facility

The components of Executive Sequence W16 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	W16	EXEC
(2)	WX16	RUNPROC

Listing of Filename = W16, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER W16
&PRI * FILE - MODFACIL
&PRI * FUNCTION - DELETE A FACILITY FROM A COURSE
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &GOTO -OUT
&SAV1 = &1
-LP2 &PRI *
&PRI * ENTER COURSE CDP (CDP), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK2
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP2
-CK2 &IF &1 EQ OUT &GOTO -OUT
&SAV2 = &1
-LP3 &PRI *
&PRI * ENTER BUILDING NO (BLDG), A5
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK3
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP3
-CK3 &IF &1 EQ OUT &GOTO -OUT
&SAV3 = &1
-LP4 &PRI *
&PRI * ENTER ROOM NO (RM), A3
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK4
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP4
-CK4 &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC WX16 &SAV1 &SAV2 &SAV3 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = WX16, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=WX16,$
END
CATALOG WX16
DELETE
FORM 4,X4,4,X4,5,X3,3
SET 3
FILE MODFACIL
&1
END
END CATALOG
```

ROOT EXECUTIVE: W17

DATA BASE: Facilities Scratch

FUNCTION: Delete an entire School

The components of Executive Sequence W17 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	W17	EXEC
(2)	WX17	RUNPROC

Listing of Filename = W17, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER W17
&PRI * FILE - MODFACIL
&PRI * FUNCTION - DELETE A SCHOOL
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC WX17 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = WX17, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=WX17,$
END
CATALOG WX17
DELETE
FORM 4
SET 1
FILE MODFACIL
&1
END
END CATALOG
```

ROOT EXECUTIVE: E38

DATA BASE: Facilities Scratch

FUNCTION: Dump Scratch data base to User's terminal

This Executive sequence produces a formatted dump to the User's terminal of all of the records of the Facilities Scratch data base.

Executive Sequence E38 Logic

The components of Executive Sequence E38 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype
(1)	E38	EXEC
(2)	RE38	RAM

Listing of Filename = E38, Filetype = EXEC

```
&TYPE OFF
&STACK KT
ATTACH RAMIS ALL BRIEF
&STACK RT
EXEC RAMISIN RE38 RAM
DETACH RAMIS BRIEF
&EXIT
```

Listing of Filename = RE38, Filetype = RAM

```
TABLE
FILE MODFACIL
PRINT BLDG AND RM AND RMCAP AND RMTP AND REQHRS AND AVHRS
BY SCH BY CDP
ON SCH SKIP-LINE
END
```

ROOT EXECUTIVE: E20

DATA BASE: Instructor Master

FUNCTION: Add a new Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file PD20/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD20/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E20 Logic

The components of Executive Sequence E20 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E20	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE20	RAM	

Listing of Filename = E20, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD20 RE20
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE20, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 9,16,2,5,4,4,4,4,4
INCLUDE 1
FILE INST
END
```

Listing of Filename = PD20, Filetype = DATA

```
0ACTION  1 1 0      ** SPECIAL PROCESSING CARD FOR INSTRUCTOR FILE **
1*
1* FUNCTION - ADD A NEW INSTRUCTOR
1*
2ID          11 9                    1                          1 9
3* ENTER INSTRUCTOR SOC SEC # (ID), A9
2NME         2116                    1                         1016
3* ENTER INSTRUCTOR SURNAME (NME), A16
2IN          31 2                    1                         26 2
3* ENTER INSTRUCTOR INITIALS (IN), A2
2RATE        41 5                    2                         28 5
3* ENTER RATE/RATING (RATE), A5
2SCH         51 4                    1                         33 4
3* ENTER SCHOOL (SCH), A4
2REP         61 4                    2                         37 4
3* ENTER REPORTING DATE (REP), A4, YYMM
2ROT         71 4                    2                         41 4
3* ENTER ROTATION DATE (ROT), A4, YYMM
2PNEC        81 4                    2                         45 4
3* ENTER PRIMARY NEC (PNEC), A4
2SNEC        91 4                    2                         49 4
3* ENTER SECONDARY NEC (SNEC), A4
```

77

ROOT EXECUTIVE: E21

DATA BASE: Instructor Master

FUNCTION: Add a new Course to an existing Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file PD21/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD21/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E21 Logic

The components of Executive Sequence E21 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E21	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE21	RAM	

Listing of Filename = E21, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD21 RE21
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE21, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 9,16,X27,4,X17,6,3,C3,C3,C3,C3,C3,C3
FORM C3,C3,C3,C3,C3,C3,C3,4,5,6
ORDER 1,2,10,13,14,14,14,14,14,14,14
ORDER 14,14,14,14,14,14,14,15,16,17
KEY 1,2,10,14
INCLUDE 10
FILE INST
END
```

Listing of Filename = PD21, Filetype = DATA

```
0ACTION  1 1 0      ** SPECIAL PROCESSING CARD FOR INSTRUCTOR FILE **
1* FUNCTION - ASSIGN A NEW COURSE
2ID          11 9                    1                          1 9
3* ENTER INSTRUCTOR SOC SEC # (ID), A9
2NME         2116                    1                         1016
3* ENTER INSTRUCTOR SURNAME (NME), A16
2CDP        101 4  XXXX              2                         53 4
3* ENTER ASSIGNED COURSE CDP (CDP), A4
```

ROOT EXECUTIVE: E55

DATA BASE: Instructor Master

FUNCTION: Change an existing Field

This single executive serves to force entry to the SCAN mode of the RAMIS data management system. After SCAN mode has been successfully entered, the User may direct change of existing field(s) interactively through his terminal.

Terminal entry requirements of the RAMIS SCAN mode are fully covered in the NCSS vendor's documentation.

The DOTS User's Manual lists the data field names and identification numbers as appropriate for the Instructor Master Data Base. This list will not be repeated in this document.

Listing of Executive E55 (SCAN Mode Entry)

```
&TYPE OFF
&STACK SCAN
&STACK FILE INST
ATTACH RAMIS ALL BR
RAMIS
DETACH RAMIS BR
&EXIT
```

ROOT EXECUTIVE: E53

DATA BASE: Instructor Master

FUNCTION: Delete a Course from an Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file PD53/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD53/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E53 Logic

The components of Executive Sequence E53 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E53	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE53	RAM	

Listing of Filename = E53, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD53 RE53
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE53, Filetype = RAM

```
DELETE
READ TRANS FILE
RECORD 250
FORM 9,16,2,X5,4,X16,4
ORDER 1,2,3,5,10
SET 10
FILE INST
END
```

Listing of Filename = PD53, Filetype = DATA

```
0ACTION  1 1 0      ** SPECIAL PROCESSING CARD FOR INSTRUCTOR FILE **
1* THIS IS EXECUTIVE NUMBER E53
1* FILE - INSTRUCTOR
1* FUNCTION - DELETE A COURSE FROM AN
1*        INSTRUCTOR THAT ALREADY EXISTS
1*        IN THE FILE...
2ID          11 9                  1                                1 9
3* ENTER INSTRUCTOR SOC SEC # (ID), A9
2NME         2116                  1                               1016
3* ENTER INSTRUCTOR SURNAME (NME), A16
2IN          31 2                  1                               26 2
3* ENTER INSTRUCTOR INITIALS (IN), A2
2SCH         51 4                  1                               33 4
3* ENTER SCHOOL (SCH), A4
2CDP        101 4  XXXX            1                               53 4
3* ENTER ASSIGNED COURSE CDP (CDP), A4
```

ROOT EXECUTIVE: E54

DATA BASE: Instructor Master

FUNCTION: Delete an Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file PD54/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD54/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E54 Logic

The components of Executive Sequence E54 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E54	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE54	RAM	

Listing of Filename = E54, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD54 RE54
&EXIT
```

Listing of Filename = ESUB1, Filetype =EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE54, Filetype = RAM

```
DELETE
READ TRANS FILE
RECORD 250
FORM 9,16,2
SET 1
FILE INST
END
```

Listing of Filename = PD54, Filetype = DATA

```
0ACTION  1 1 0      ** SPECIAL PROCESSING CARD FOR INSTRUCTOR FILE **
1* THIS IS EXECUTIVE NUMBER E54
1* FILE - INSTRUCTOR
1* FUNCTION - DELETE AN INSTRUCTOR
1*        THAT ALREADY EXISTS
1*        IN THE FILE...
2ID          11 9                  1                          1 9
3* ENTER INSTRUCTOR SOC SEC # (ID), A9
2NME         2116                  1                         1016
3* ENTER INSTRUCTOR SURNAME (NME), A16
2IN          31 2                  1                         26 2
3* ENTER INSTRUCTOR INITIALS (IN), A2
2SCH         51 4                  1                         33 4
3* ENTER SCHOOL (SCH), A4
```

ROOT EXECUTIVE: E23

DATA BASE: Instructor Master

FUNCTION: Load the Instructor Master data base from card input

The Instructor Master data base is loaded from scratch by using the following Executive Sequences:

Executive Sequence	Input Card Format	Input Data Contents
E23	Instructor File, TYPE -1- Input Card	Instructor Fixed Data
E24	Instructor File, TYPE -2- Input Card	Instructor Assignment Data

In the normal loading sequence, Executive E23 must have been successfully executed before attempting Executive E24 operations.

The format of the card input data set associated with Executive Sequence E23 is illustrated in the diagram following entitled "Instructor Data Base Type -1- Input Card Format".

Executive Sequence E23 Logic

The components of Executive Sequence E23 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	E23	EXEC
(2)	RE23	RAM

Listing of Filename = E23, Filetype = EXEC

```
&TYPE OFF
STATE E23 DATA
&IF &INDEX0 NE 0 &GOTO -NODATA
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE23 RAM
DETACH RAMIS BR
ERASE E23 DATA
&EXIT
-NODATA &PRINT NO TYPE1 CARDS PROVIDED -- FUNCTION TERMINATING
```

Listing of Filename = RE23, Filetype = RAM

```
ERASE
FILE INST
END
REVISE
READ E23 DATA
RECORD 80
FORM 9,16,2,5,4,4,4,4,4
KEY 1
INCLUDE 1
FILE INST
END
```

INSTRUCTOR DATA BASE TYPE -1- INPUT CARD FORMAT

INSTRUCTOR FILE

IBM Federal Systems Division, Space Systems Center, Huntsville, Alabama

NAME: INSTRUCTOR TIME CARD

JOB NO | FORM PACKAGE | WRITTEN AS | PUNCH AS | CARD PUNCH PRINT YES NO | VERIFY YES NO | DATE DUE OUT | SHEET OF

80-Column Card Punch Layout

ID	SCH	CDP	A1	A2	A3	A4	A5	A6	A7	B1	B2	B3	B4	B5	B6	B6A
			INSTRUCTOR TIME(S), AS REQUIRED = XX$_\wedge$X													

$\wedge$ = LOCATION OF ASSUMED DECIMAL POINT

955-1115-0

ROOT EXECUTIVE: E24

DATA BASE: Instructor Master

FUNCTION: Load the Instructor Master data base from card input

The Instructor Master data base is loaded from scratch by using the following Executive Sequences:

Executive Sequence	Input Card Format	Input Data Contents
E23	Instructor File, TYPE -1- Input Card	Instructor Fixed Data
E24	Instructor File, TYPE -2- Input Card	Instructor Assignment Data

In the normal loading sequence, Executive E23 must have been successfully executed before attempting Executive E24 operations.

The format of the card input data set associated with Executive Sequence E24 is illustrated in the diagram following entitled "Instructor Data Base TYPE -2- Input Card Format".

Executive Sequence E24 Logic

The components of Executive Sequence E24 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E24	EXEC	
(2)	PGM9B	PLI	(PL/1 Program)
(3)	RE24	RAM	

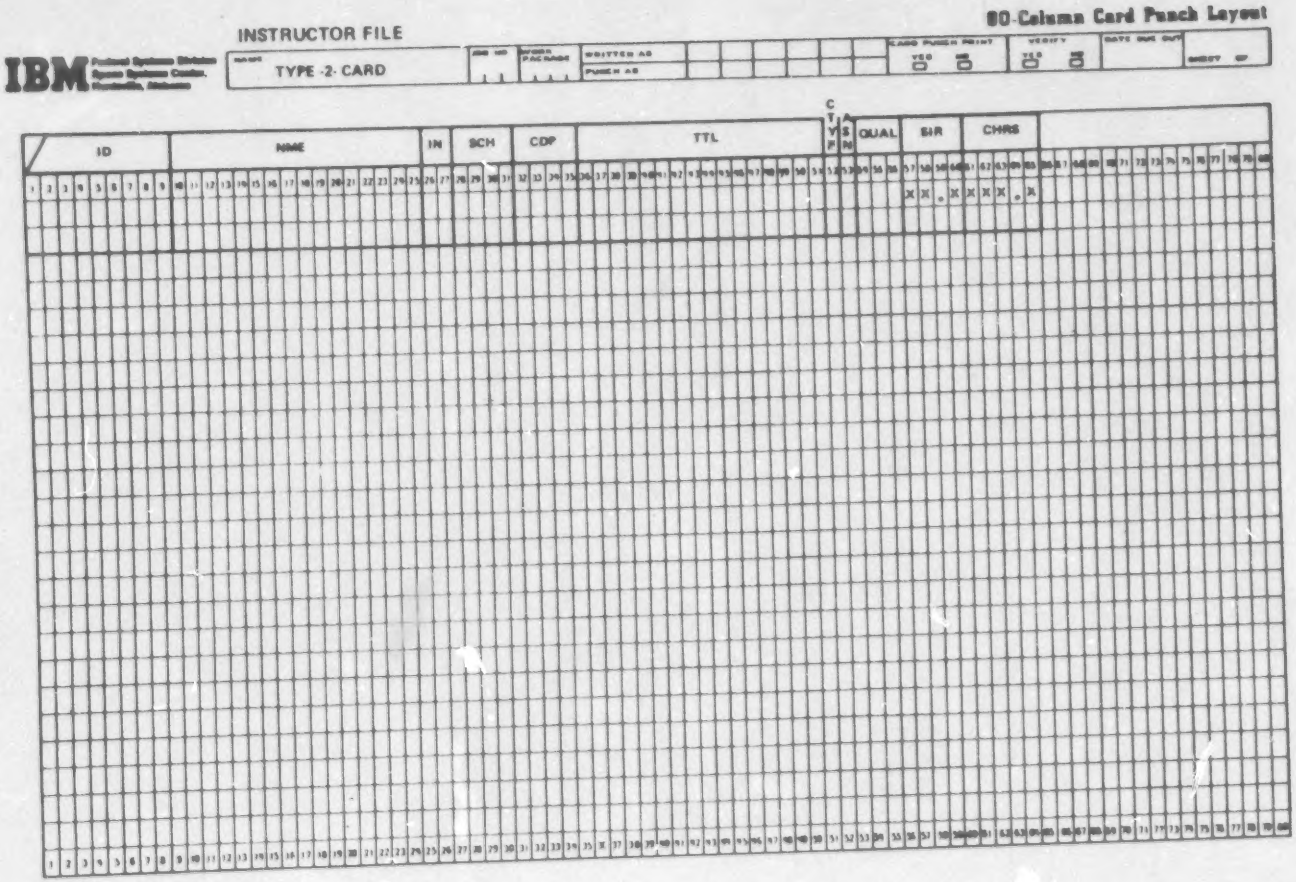
Listing of Filename = E24, Filetype = EXEC

```
&TYPE OFF
STATE E24 DATA
&IF &INDEX0 NE 0 &GOTO -NOCDS
FILEDEF CARDS DSK E24 DATA
FILEDEF TRANS DSK TYPE2 DATA LR 250 RE F
LOAD PGM9B (CLEAR LIBE) PL1LIB
START (BRIEF)
FILEDEF CARDS CLEAR
FILEDEF TRANS CLEAR
ERASE E24 DATA
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE24 RAM
DETACH RAMIS BR
&EXIT
-NOCDS &PRINT NO TYPE2 CARDS FOUND -- FUNCTION TERMINATING
&EXIT
```

Listing of Filename = RE24, Filetype = RAM

```
REVISE
READ TYPE2 DATA
RECORD 250
FORM 9,16,2,4,4,16,3,C3,C3,C3,C3,C3,C3,C3,C3,C3,C3,C3,C3,C3,4,5,6
ORDER 1,2,3,5,10,11,14,14,14,14,14,14,14,14,14,14,14,14,14,14,15,16,17
KEY 1,2,3,5,10,14
INCLUDE 10
FILE INST
END
```

INSTRUCTOR (COURSE) DATA BASE TYPE -2- INPUT CARD FORMAT



ROOT EXECUTIVE: W20

DATA BASE: Instructor Scratch

FUNCTION: Add a new Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD20/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD20/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W20 Logic

The components of Executive Sequence W20 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	W20	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW20	RAM	

Listing of Filename = W20, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD20 RW20
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW20, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 9,16,2,5,4,4,4,4,4
INCLUDE 1
FILE MODINST
END
```

Listing of Filename = WPD20, Filetype = DATA

```
0ACTION  1 1 0      ** SPECIAL PROCESSING CARD FOR INSTRUCTOR FILE *
1* THIS IS EXECUTIVE NUMBER W20
1* FILE - INSTRUCTOR SCRATCH DATA BASE
1* FUNCTION - ADD A NEW INSTRUCTOR PLUS
1*             INSTRUCTOR FIXED DATA...
2ID          11 9                  1                          1 9
3* ENTER INSTRUCTOR SOC SEC # (ID), A9
2NME         2116                  1                         1016
3* ENTER INSTRUCTOR SURNAME (NME), A16
2IN          31 2                  1                         26 2
3* ENTER INSTRUCTOR INITIALS (IN), A2
2RATE        41 5                  2                         28 5
3* ENTER RATE/RATING (RATE), A5
2SCH         51 4                  1                         33 4
3* ENTER SCHOOL (SCH), A4
2REP         61 4                  2                         37 4
3* ENTER REPORTING DATE (REP), A4, YYMM
2ROT         71 4                  2                         41 4
3* ENTER ROTATION DATE (ROT), A4, YYMM
2PNEC        81 4                  2                         45 4
3* ENTER PRIMARY NEC (PNEC), A4
2SNEC        91 4                  2                         49 4
3* ENTER SECONDARY NEC (SNEC), A4
```

ROOT EXECUTIVE: W53

DATA BASE: Instructor Scratch

FUNCTION: Delete a Course from an Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD53/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD53/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W53 Logic

The components of Executive Sequence W53 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	W53	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW53	RAM	

Listing of Filename = W53, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD53 RW53
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW53, Filetype = RAM

```
DELETE
READ TRANS FILE
RECORD 250
FORM 9,16,2,X5,4,X16,4
ORDER 1,2,3,5,10
SET 10
FILE MODINST
END
```

Listing of Filename = WPD53, Filetype = DATA

```
0ACTION  1 1 0      ** SPECIAL PROCESSING CARD FOR INSTRUCTOR FILE **
1* THIS IS EXECUTIVE NUMBER W53
1* FILE - INSTRUCTOR SCRATCH DATA BASE
1* FUNCTION - DELETE A COURSE FROM AN
1*        INSTRUCTOR THAT ALREADY EXISTS
1*        IN THE FILE...
2ID          11 9                  1                          1 9
3* ENTER INSTRUCTOR SOC SEC # (ID), A9
2NME         2116                  1                         1016
3* ENTER INSTRUCTOR SURNAME (NME), A16
2IN          31 2                  1                         26 2
3* ENTER INSTRUCTOR INITIALS (IN), A2
2SCH         51 4                  1                         33 4
3* ENTER SCHOOL (SCH), A4
2CDP        101 4  XXXX            1                         53 4
3* ENTER ASSIGNED COURSE CDP (CDP), A4
```

ROOT EXECUTIVE: W21

DATA BASE: Instructor Scratch

FUNCTION: Add a new Course to an existing Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD21/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD21/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W21 Logic

The components of Executive Sequence W21 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	W21	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW21	RAM	

Listing of Filename = W21, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD21 RW21
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW21, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 9,16,2,X5,4,X16,4,16,1,6,3,C3,C3,C3,C3,C3,C3
FORM C3,C3,C3,C3,C3,C3,C3,4,5,6
ORDER 1,2,3,5,10,11,12,13,14,14,14,14,14,14,14
ORDER 14,14,14,14,14,14,14,15,16,17
KEY 1,2,3,5,10,14
INCLUDE 10
FILE MODINST
END
```

Listing of Filename = WPD21, Filetype = DATA

```
0ACTION  1 1 0      ** SPECIAL PROCESSING CARD FOR INSTRUCTOR FILE **
1* THIS IS EXECUTIVE NUMBER W21
1* FILE - INSTRUCTOR SCRATCH DATA BASE
1* FUNCTION - ASSIGN A NEW COURSE TO AN
1*        INSTRUCTOR THAT ALREADY EXISTS
1*        IN THE FILE...
2ID          11 9                    1                                  1 9
3* ENTER INSTRUCTOR SOC SEC # (ID), A9
2NME         2116                    1                                 1016
3* ENTER INSTRUCTOR SURNAME (NME), A16
2IN          31 2                    1                                 26 2
3* ENTER INSTRUCTOR INITIALS (IN), A2
2SCH         51 4                    1                                 33 4
3* ENTER SCHOOL (SCH), A4
2CDP        101 4  XXXX              2                                 53 4
3* ENTER ASSIGNED COURSE CDP (CDP), A4
2TTL        11116                    2                                 5716
3* ENTER COURSE TITLE (TTL), A16
2ONB        121 1  1                 2                                 73 1
3* ENTER ONBOARD FLAG (ONB), A1
3*        1 = YES, 2 = NO
```

ROOT EXECUTIVE: W54

DATA BASE: Instructor Scratch

FUNCTION: Delete an Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD54/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD54/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W54 Logic

The components of Executive Sequence W54 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	W54	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW54	RAM	

Listing of Filename = W54, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD54 RW54
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW54, Filetype = RAM

```
DELETE
READ TRANS FILE
RECORD 250
FORM 9,16,2
SET 1
FILE MODINST
END
```

Listing of Filename = WPD54, Filetype = DATA

```
0ACTION  1 1 0      ** SPECIAL PROCESSING CARD FOR INSTRUCTOR FILE **
1* THIS IS EXECUTIVE NUMBER W54
1* FILE - INSTRUCTOR SCRATCH DATA BASE
1* FUNCTION - DELETE AN INSTRUCTOR
1*        THAT ALREADY EXISTS
1*        IN THE FILE...
2ID          11 9                    1                                1 9
3* ENTER INSTRUCTOR SOC SEC # (ID), A9
2NME         2116                    1                               1016
3* ENTER INSTRUCTOR SURNAME (NME), A16
2IN          31 2                    1                               26 2
3* ENTER INSTRUCTOR INITIALS (IN), A2
2SCH         51 4                    1                               33 4
3* ENTER SCHOOL (SCH), A4
```

ROOT EXECUTIVE: W55

DATA BASE: Instructor Scratch

FUNCTION: Change an existing Field

This single executive serves to force entry to the SCAN mode of the RAMIS data management system. After SCAN mode has been successfully entered, the user may direct change of existing field(s) interactively through his terminal.

Terminal entry requirements of the RAMIS SCAN mode are fully covered in the NCSS vendor's documentation.

The DOTS User's Manual lists the data field names and identification numbers as appropriate for the Instructor Scratch Data Base. This list will not be repeated in this document.

Listing of Executive W55 (SCAN Mode Entry)

```
&TYPE OFF
&STACK KT
&STACK SCAN
&STACK FILE MODINST
ATTACH RAMIS ALL BR
&STACK RT
RAMIS
DETACH RAMIS BR
&EXIT
```

ROOT EXECUTIVE: E32

DATA BASE: Instructor Master
Instructor Scratch

FUNCTION: Recreate the Scratch data base using the records from one User-selected school in the Master data base

The first step in the recreation of the Scratch data base is the erase of the existing Scratch copy.

After being prompted, the User must supply the 4-character name of the school of interest.

The Master data base is then dumped into card-image format onto a temporary file, selecting only those records associated with the school chosen by the User. The temporary file is then used as input to recreate the Scratch data base in its entirety.

At the successful conclusion of this Executive Sequence the contents of the Master data base are unchanged. The new Scratch data base will contain only the records associated with the selected school.

Executive Sequence E32 Logic

The components of Executive Sequence E32 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E32	EXEC	
(2)	E57	EXEC	
(3)	X57	RUNPROC	
(4)	E29	EXEC	
(5)	PGM7	PLI	(PL/1 Program)
(6)	RE29	RAM	

Listing of Filename = E32, Filetype = EXEC

```
&TYPE OFF
EXEC E57
EXEC E29
```

Listing of Filename = E57, Filetype = EXEC

```
&TYPE OFF
&PRINT * FILE - INSTRUCTOR
&PRINT * FUNCTION - SELECT INSTRUCTORS BY SCHOOL
&PRINT * ENTER OUT FOR PREMATURE EXIT
-LP1 &PRINT *
&PRINT * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &EXIT
FILEDEF RAMUTIL1 DSK INST DUMP LR 80 RE FB BL 800
&STACK KT
&STACK EXEC X57 &1
&STACK FIN
ATTACH RAMIS ALL BR
FILEDEF 6 MSG KILL
EXEC RAMIS
DETACH RAMIS BR
&STACK RT
FILEDEF RAMUTIL1 CLEAR
FILEDEF 6 CONO
&STACK KT
&STACK L /D=/
&STACK Q
EDIT MSG KILL
&PRI &1 SCHOOL DUMPED TO DISK FILE
&EXIT
-ERR1 &PRINT *** INVALID ENTRY -- FUNCTION TERMINATING
```

Listing of Filename = X57, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X57,$
END
CATALOG X57
DUMP
FILE INST
PRINT YTD AND SAVE
IF SCH IS &1
END
END CATALOG
```

Listing of Filename = E29, Filetype = EXEC

```
&TYPE OFF
FILEDEF TRANS DSK REFOR INST LR 97 RE F
FILEDEF DUMP DSK INST DUMP
LOAD PGM7 (CLEAR LIBE) PLILIB
START (BRIEF)
FILEDEF TRANS CLEAR
FILEDEF DUMP CLEAR
&STACK KT
ATTACH RAMIS ALL BR
FILEDEF 6 MSG KILL
EXEC RAMISIN RE29 RAM
DETACH RAMIS BR
EXEC RAMOUT
&EXIT
```

Listing of Filename = RE29, Filetype = RAM

```
ERASE
FILE MODINST
END
INPUT
READ REFOR INST
RECORD 97
FORM 9,16,2,5,4,4,4,4,4,4,16,1,6,3,4,5,6
SET 1
SEARCH FORWARD
FILE MODINST
END
```

ROOT EXECUTIVE: E27

DATA BASE: Instructor Master
Instructor Scratch

FUNCTION: Recreate the Scratch data base using the records from all Schools in the Master data base

The first step in the recreation of the Scratch data base is the erase of the existing Scratch copy.

The Master is then dumped into card-image format onto a temporary dataset. This file is then used as input to recreate the Scratch data base in its entirety.

Executive Sequence E27 Logic

The components of Executive Sequence E27 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E27	EXEC	
(2)	RE25	RAM	
(3)	PGM7	PLI	(PL/1 Program)
(4)	RE27	RAM	

Listing of Filename = E27, Filetype = EXEC

```
&TYPE OFF
ATTACH TEMP20 AS 193 B
FILEDEF RAMUTIL1 DSK UINST DUMP U LR 80 RE FB BL 800
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE25 RAM
DETACH RAMIS BR
FILEDEF RAMUTIL1 CLEAR
FILEDEF TRANS DSK UREFOR INST U LR 97 RE FB BL 776
FILEDEF DUMP DSK UINST DUMP U
LOAD PGM7 (CLEAR LIBE) PLILIB
START (BRIEF)
FILEDEF TRANS CLEAR
FILEDEF DUMP CLEAR
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE27 RAM
DETACH RAMIS BR
DETACH 193
```

Listing of Filename = RE25, Filetype = RAM

```
DUMP
FILE INST
PRINT YTD AND SAVE
END
```

Listing of Filename = RE27, Filetype = RAM

```
ERASE
FILE MODINST
END
INPUT
READ UREFOR INST
RECORD 97
FORM 9,16,2,5,4,4,4,4,4,4,16,1,6,3,4,5,6
SET 1
SEARCH FORWARD
FILE MODINST
END
```

ROOT EXECUTIVE: E30

DATA BASE: Instructor Master

FUNCTION: Dump the records of one (1) School to the User's terminal

This Executive sequence forces a formatted dump to the User's terminal of all of the records of the Instructor Master data base associated with one school. The Executive will prompt the User for the selection of the desired school.

Executive Sequence E30 Logic

The components of Executive Sequence E30 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E30	EXEC	
(2)	E57	EXEC	
(3)	X57	RUNPROC	
(4)	E60	EXEC	
(5)	PGM8	PLI	(PL/1 Program)

Listing of Filename = E30, Filetype = EXEC

```
&TYPE OFF
EXEC E57
EXEC E60
```

Listing of Filename = E57, Filetype = EXEC

```
&TYPE OFF
&PRINT * FILE - INSTRUCTOR
&PRINT * FUNCTION - SELECT INSTRUCTORS BY SCHOOL
&PRINT * ENTER OUT FOR PREMATURE EXIT
-LP1 &PRINT *
&PRINT * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &EXIT
FILEDEF RAMUTIL1 DSK INST DUMP LR 80 RE FB BL 800
&STACK KT
&STACK EXEC X57 &1
&STACK FIN
ATTACH RAMIS ALL BR
FILEDEF 6 MSG KILL
EXEC RAMIS
DETACH RAMIS BR
&STACK RT
FILEDEF RAMUTIL1 CLEAR
FILEDEF 6 CONO
&STACK KT
&STACK L /D=/
&STACK Q
EDIT MSG KILL
&PRI &1 SCHOOL DUMPED TO DISK FILE
&EXIT
-ERR1 &PRINT *** INVALID ENTRY -- FUNCTION TERMINATING
```

Listing of Filename = X57, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X57,$
END
CATALOG X57
DUMP
FILE INST
PRINT YTD AND SAVE
IF SCH IS &1
END
END CATALOG
```

Listing of Filename = E60, Filetype = EXEC

```
&TYPE OFF
FILEDEF DUMP DSK INST DUMP LR 80 RE F
LOAD PGM8 (CLEAR LIBE) PLILIB
START (BRIEF)
FILEDEF DUMP CLEAR
&EXIT
```

ROOT EXECUTIVE: E35

DATA BASE: Instructor Scratch

FUNCTION: Dump Records of one School to User's terminal

This Executive sequence forces a formatted dump to the User's terminal of all of the records of the Instructor Scratch data base associated with one school. The Executive will prompt the User for the selection of the desired school.

Executive Sequence E35 Logic

The components of Executive Sequence E35 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E35	EXEC	
(2)	E28	EXEC	
(3)	X28	RUNPROC	
(4)	E60	EXEC	
(5)	PGM8	PLI	(PL/1 Program)

Listing of Filename = E35, Filetype = EXEC

```
&TYPE OFF
EXEC E28
EXEC E60
```

Listing of Filename = E28, Filetype = EXEC

```
&TYPE OFF
&PRINT * FILE - INSTRUCTOR SCRATCH DATA BASE
&PRINT * FUNCTION - SELECT INSTRUCTORS BY SCHOOL
&PRINT * ENTER OUT FOR PREMATURE EXIT
-LP1 &PRINT *
&PRINT * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &EXIT
FILEDEF RAMUTIL1 DSK INST DUMP LR 80 RE FB BL 800
&STACK KT
&STACK EXEC X28 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
FILEDEF RAMUTIL1 CLEAR
&EXIT
-ERR1 &PRINT *** INVALID ENTRY -- FUNCTION TERMINATING
```

Listing of Filename = X28, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X28.$
END
CATALOG X28
DUMP
FILE MODINST
PRINT YTD AND SAVE
IF SCH IS &1
END
END CATALOG
```

Listing of Filename = E60, Filetype = EXEC

```
&TYPE OFF
FILEDEF DUMP DSK INST DUMP LR 80 RE F
LOAD PGM8 (CLEAR LIBE) PLILIB
START (BRIEF)
FILEDEF DUMP CLEAR
&EXIT
```

ROOT EXECUTIVE: E22A

DATA BASE: Instructor

FUNCTION: Sort the Instructor time card file

All Root Executives with names containing the E22-prefix are used in conjunction with the processing of the weekly Instructor Time Cards.

The User will be prompted for the date of the time card file desired. This date takes the format Wyymmdd; in this date 'yy' is the year, 'mm' the month, and 'dd' the day. The entry will be a total of seven (7) characters in length; provide a leading zero on the month and/or day, if applicable, e.g., W750911. This keyboard entry becomes the "filetype" for the time card file desired, i.e., the input file to the sort is filename = TCARDS, filetype = Wyymmdd.

The sorted output file of this Executive is filename = E22, filetype = SORT.

The components of Executive Sequence E22A and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	E22A	EXEC

Listing of Filename = E22A, Filetype = EXEC

```
&TYPE OFF
&PRINT * ENTER OUT TO ABORT
-STEP1 &PRINT *
&PRINT * ENTER FILE-TYPE (FT) OF THE TIME CARD FILE
&PRINT *       TO BE PROCESSED; FORMAT IS WYYMMDD
&PRINT *
&READ ARGS
&IF &INDEX EQ 1 &GOTO -STEP2
&PRINT * WRONG NUMBER ARGUMENTS ????? TRY AGAIN
&GOTO -STEP1
-STEP2 &IF &1 EQ OUT &EXIT
&PARM = &1
STATE TCARDS &PARM P
&IF &INDEX0 EQ 0 &GOTO -STEP3
&PRINT * FILE TCARDS &PARM DOES NOT EXIST!
&PRINT * EXECUTIVE E22A TERMINATING
 &EXIT
-STEP3 ATTACH TEMP5 AS 193 B
ERASE E22 SORT P
FILEDEF SORTOUT DSK E22 SORT LR 80 RE FB BL 800
OSSORT SRT TCARDS &PARM P E22 SORT P (WMODE=U MSG CC)
&PRINT * SORT RETURN CODE WAS &INDEX0
DETACH 193 BR
FILEDEF SORTOUT CLEAR
ERASE TCARDS &PARM P
COMBINE TCARDS &PARM P E22 SORT P
```

ROOT EXECUTIVE: E22B

DATA BASE: Instructor

FUNCTION: Makeup the Instructor data base reference file, Filename = SPIN, Filetype - DATA

All Root Executives with names containing the E22-prefix are used in conjunction with the processing of the weekly Instructor Time Cards.

The components of Executive Sequence E22B and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	E22B	EXEC

Listing of Filename = E22B, Filetype = EXEC

```
&TYPE OFF
&STACK KT
ERASE SPIN DATA
FILEDEF FT04F001 DSK FILE FT04F001 LR 17 RE FB BL 799
&BEGSTACK
TABLEF
FILE INST
PRINT ID AND SCH AND CDP AND SAVE
END
FIN
&ENDSTACK
ATTACH RAMIS ALL BR
RAMIS
DETACH RAMIS BR
FILEDEF FT04F001 CLEAR
ALTER FILE FT04F001 P SPIN DATA P
&PRINT *
&PRINT * FILE SPIN DATA NOW EXISTS
```

ROOT EXECUTIVE: E22C

DATA BASE: Instructor

FUNCTION: Compare the Instructor Time Card input file with the, Instructor data base reference file

All Root Executives with names containing the E22-prefix are used in conjunction with the processing of the weekly Instructor Time Cards.

In this sequence, the sorted file containing the weekly Instructor Time Cards (E22/SORT) is compared with the Instructor data base reference file (SPIN/DATA). Any discrepancies are noted in the form of messages in output file (E22/EMSG). Time cards that contain no errors are passed to output file E22/DATA. Time cards with errors are placed in the file E22/BUST.

The input files E22/SORT and SPIN/DATA remain unchanged as a result of the operation of this Executive Sequence.

The components of Executive Sequence E22C and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E22C	EXEC	
(2)	PGM22	PLI	(PL/1 Program)

Listing of Filename = E22C, Filetype = EXEC

```
&TYPE OFF
ERASE E22 DATA
ERASE E22 BUST
ERASE E22 EMSG
FILEDEF CARDS DSK E22 SORT LR 80 RE FB BL 800
FILEDEF SPIN DSK SPIN DATA LR 17 RE FB BL 799
FILEDEF GOOD DSK E22 DATA LR 80 RE FB BL 800
FILEDEF ETRANS DSK E22 BUST LR 80 RE FB BL 800
FILEDEF EMSG DSK E22 EMSG LR 61 RE FB BL 793
LOAD PGM22 (CLEAR LIBL) PLILIB
START (BRIEF)
&PRINT *
STATE E22 EMSG
&IF &INDEX0 NE 0 &GOTO -STEP1
&PRINT * E22 EMSG FILE EXISTS...
&GOTO -STEP2
-STEP1 &PRINT *E22 EMSG ERROR FILE DOES NOT EXIST...
-STEP2 &PRINT *
STATE E22 DATA
&IF &INDEX0 NE 0 &GOTO -STEP3
&PRINT * E22 DATA FILE NOW EXISTS...
&GOTO -STEP4
-STEP3 &PRINT * E22 DATA FILE DOES NOT EXIST...
-STEP4 FILEDEF CARDS CLEAR
FILEDEF SPIN CLEAR
FILEDEF GOOD CLEAR
FILEDEF ETRANS CLEAR
FILEDEF EMSG CLEAR
```

ROOT EXECUTIVE: E22D

DATA BASE: Instructor

FUNCTION: Zero the Current Week (CW) time fields within the Instructor data base

All Root Executives with names containing the E22-prefix are used in conjunction with the processing of the weekly Instructor Time Cards.

This Executive Sequence clears the time entries to the Current Week (CW) fields (previous weeks' entries) for all Instructors.

The components of Executive Sequence E22D and the chronological order of execution are indicated in the table presented below. A listing of these files follows:

Sequence	Filename	Filetype
(1)	E22D	EXEC

Listing of Filename = E22D, Filetype = EXEC

```
&TYPE OFF
&STACK KT
FILEDEF 6 MSG KILL
&BEGSTACK
SCAN
FILE INST
SHOW. 15
CHANGE. 15=>0,$*
END
FIN
&ENDSTACK
ATTACH RAMIS ALL BR
EXEC RAMIS
DETACH RAMIS ALL BR
FILEDEF 6 CONO
```

ROOT EXECUTIVE: E22E

DATA BASE: Instructor

Function: Load the Current Week's Instructor Time Cards into the Instructor data base

All Root Executives with names containing the E22-prefix are used in conjunction with the processing of the weekly Instructor Time Cards.

Input time card file to this Executive Sequence is filename = E22, filetype = DATA.

The components of Executive Sequence E22E and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E22E	EXEC	
(2)	PGM5	PL1	(PL/1 Program)
(3)	RE22	RAM	

Listing of Filename = E22E, Filetype = EXEC

```
&TYPE OFF
STATE E22 DATA
&IF &INDEX0 NE 0 &GOTO -NOCDS
ERASE TIME REJECTS
FILEDEF CARDS DSK E22 DATA
FILEDEF TRANS DSK TIME FILE LR 121 RE F
FILEDEF ETRANS DSK TIME REJECTS LR 80 RE FB BL 800
LOAD PGM5 (CLEAR LIBE) PLILIB
START (BRIEF)
FILEDEF CARDS CLEAR
FILEDEF TRANS CLEAR
FILEDEF ETRANS CLEAR
ERASE REJECT FILE
FILEDEF 09 DSK REJECT FILE LR 121 RE F
&STACK KT
ATTACH RAMIS ALL BR
FILEDEF 6 MSG KILL
EXEC RAMISIN RE22 RAM
DETACH RAMIS BR
EXEC RAMOUT
FILEDEF 09 CLEAR
STATE REJECT FILE
&IF &INDEX0 EQ 0 &GOTO -STP1
&PRINT * REJECT FILE DOES NOT EXIST
ERASE E22 SORT
ERASE E22 DATA
&EXIT
-STP1 &PRINT * REJECT FILE EXISTS: ERRORS TO BE RESOLVED
&EXIT
```

Listing of Filename = RE22, Filetype = RAM

```
UPDATE
READ TIME FILE
RECORD 121
SET 13,15,16,17
NO-KEY 09
FORM 9,4,4,6,3,4,X-4,4,X-4,4,C3,C4,X-4,C4,X-4,C4,C3,C4,X-4,C4,X-4,C4
FORM C3,C4,X-4,C4,X-4,C4,C3,C4,X-4,C4,X-4,C4,C3,C4,X-4,C4,X-4,C4
FORM C3,C4,X-4,C4,X-4,C4,C3,C4,X-4,C4,X-4,C4,C3,C4,X-4,C4,X-4,C4
FORM C3,C4,X-4,C4,X-4,C4,C3,C4,X-4,C4,X-4,C4,C3,C4,X-4,C4,X-4,C4
FORM C3,C4,X-4,C4,X-4,C4,C3,C4,X-4,C4,X-4,C4
ORDER 1,5,10,13,14,15,16,17,14,15,16,17,14,15,16,17,14,15,16,17
ORDER 14,15,16,17,14,15,16,17,14,15,16,17,14,15,16,17,14,15,16,17
ORDER 14,15,16,17,14,15,16,17,14,15,16,17,14,15,16,17,14,15,16,17
FILE INST
END
```

ROOT EXECUTIVE: E22F

DATA BASE: Instructor

FUNCTION: Clear all Quarter-to-Date (QTD) time field entries for all Instructors

All Root Executives with names containing the E22-prefix are used in conjunction with the processing of the weekly Instructor Time Cards.

After processing any desired quarterly reports, this Executive Sequence serves to zero all accumulated quarterly times.

The components of Executive Sequence E22F and the chronological order of execution are indicated in the table presented below. A listing of this file follows:

Sequence	Filename	Filetype
(1)	E22F	EXEC

Listing of Filename = E22F, Filetype = EXEC

```
&TYPE OFF
&STACK KT
FILEDEF 6 MSG KILL
&BEGSTACK
SCAN
FILE INST
SHOW. 16
CHANGE. 16=>0,$*
END
FIN
&ENDSTACK
ATTACH RAMIS ALL BR
EXEC RAMIS
DETACH RAMIS ALL BR
FILEDEF 6 CONO
```

ROOT EXECUTIVE: E22G

DATA BASE: Instructor

FUNCTION: Zero (Clear all Year-to-Date (YTD) time field entries for all Instructors

All Root Executives with names containing the E22-prefix are used in conjunction with the processing of the weekly Instructor Time Cards.

After processing any desired quarterly and yearly reports, this Executive Sequence serves to zero all accumulated yearly times.

The components of Executive Sequence E22G and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	E22G	EXEC

Listing of Filename = E22G, Filetype = EXEC

```
&TYPE OFF
&STACK KT
FILEDEF 6 MSG KILL
&BEGSTACK
SCAN
FILE INST
SHOW. 17
CHANGE. 17=>0,$*
END
FIN
&ENDSTACK
ATTACH RAMIS ALL BR
EXEC RAMIS
DETACH RAMIS ALL BR
FILEDEF 6 CONO
```

ROOT EXECUTIVE: E22R1

DATA BASE: Instructor

FUNCTION: Produce a standard report from data within the Instructor data base

The components of Executive Sequence E22R1 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	E22R1	EXEC
(2)	RE22R1	RUNPROC

Listing of Filename = E22R1, Filetype = EXEC

```
&TYPE OFF
&STACK KT
&STACK EXEC RE22R1
ATTACH RAMIS ALL BR
RAMIS
DETACH RAMIS BR
&STACK RT
&EXIT
```

Listing of Filename = RE22R1, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=RE22R1,$
END
CATALOG RE22R1
&&   PRINT * ENTER OUT TO ABORT
&&STEP1
&&   PRINT *
&&   PRINT * ENTER INSTRUCTOR ID (ID), A9
&&   READ ARGS
&&   IF &NARGS EQ 1 GOTO STEP2 ;
&&   PRINT * ???? WRONG NUMBER ARGUMENTS - TRY AGAIN
&&   GOTO STEP1
&&STEP2
&&   IF &1 NE 'OUT' GOTO STEP3 ;
&&   EXIT
&&STEP3
TABLE
HEADING
     GENERAL QUERY OF INSTRUCTOR ASSIGNMENTS
     AGAINST FOLLOWING INSTRUCTOR ID(S):
     &1
FILE INST
PRINT CDP
     BY SCH BY ID BY IN BY NME
ON ID SKIP-LINE
IF CDP IS-NOT ' '
IF ID IS &1
END
FIN
END CATALOG
```

ROOT EXECUTIVE: E22R2

DATA BASE: Instructor

FUNCTION: Produce a standard report from data within the Instructor data base

The components of Executive Sequence E22R2 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	E22R2	EXEC
(2)	E22T1	TABLE

Listing of Filename = E22R2, Filetype = EXEC

```
&TYPE OFF
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN E22T1 TABLE
DETACH RAMIS BR
&EXIT
```

Listing of Filename = E22T1, Filetype = TABLE

```
TABLE
HEADING
     GENERAL-PURPOSE INSTRUCTOR ASSIGNMENT
     LISTING FOR TIME-CARD VALIDATION...
FILE INST
PRINT NME AND CDP
BY SCH BY ID
IF CDP IS-NOT ' '
IF SCH IS DCFF
ON ID SKIP-LINE
END
```

ROOT EXECUTIVE: E22R3

DATA BASE: Instructor

FUNCTION: Produce a standard report from data within the Instructor data base

The components of Executive Sequence E22R3 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	E22R3	EXEC
(2)	SELSCH	EXEC
(3)	RE22R3	RUNPROC

Listing of Filename = E22R3, Filetype = EXEC

```
&TYPE OFF
EXEC SELSCH RE22R3
&EXIT
```

Listing of Filename = SELSCH, Filetype = EXEC

```
&TYPE OFF
&WHICH = &1
&PRINT *
&PRINT * ENTER OUT TO ABORT
-LP1 &PRINT *
&PRINT * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRINT * ???? WRONG NUMBER PARMS - TRY AGAIN
&GOTO -LP1
-CK1 &IF &1 EQ OUT &EXIT
&STACK KT
&STACK EXEC &WHICH &1
&STACK FIN
ATTACH RAMIS ALL BR
RAMIS
DETACH RAMIS BR
&STACK RT
&EXIT
```

Listing of Filename = RE22R3, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=RE22R3,$
END
CATALOG RE22R3
DEFINE
FILE INST
SCW/F8.1 = CW ;
END
TABLE
HEADING
     WEEKLY INSTRUCTOR TIME SUMMARY FOR
     TIME-CARD VALIDATION...
FILE INST
SUM SCW AS 'EXPENDED,TIME'
     BY SCH AS SCHOOL
     BY ID
     BY IN
     BY NME AS NAME
IF TT IS-NOT ' '
IF CDP IS-NOT ' '
IF SCH IS &1
ON ID SKIP-LINE
END
END CATALOG
```

ROOT EXECUTIVE: E22R4

DATA BASE: Instructor

FUNCTION: Produce a standard report from data within the Instructor data base

The components of Executive Sequence E22R4 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	E22R4	EXEC
(2)	RE22R4	RUNPROC

Listing of Filename = E22R4, Filetype = EXEC

```
&TYPE OFF
&STACK KT
&STACK EXEC RE22R4
ATTACH RAMIS ALL BR
RAMIS
DETACH RAMIS BR
&STACK RT
&EXIT
```

Listing of Filename = RE22R4, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=RE22R4,$
END
CATALOG RE22R4
&&   PRINT * ENTER OUT TO ABORT
&&STEP1
&&   PRINT *
&&   PRINT * ENTER INSTRUCTOR ID (ID), A9
&&   READ ARGS
&&   IF &NARGS EQ 1 GOTO STEP2 ;
&&   PRINT * ???? WRONG NUMBER ARGUMENTS - TRY AGAIN
&&   GOTO STEP1
&&STEP2
&&   IF &1 NE 'OUT' GOTO STEP3 ;
&&   EXIT
&&STEP3
DEFINE
FILE INST
SCW/F8.1 = CW ;
END
TABLE
HEADING
     WEEKLY INSTRUCTOR INDIVIDUAL TIME
     DISTRIBUTION AMONG COURSES...
FILE INST
SUM SCW AS 'EXPENDED,TIME'
     BY SCH AS SCHOOL
     BY ID SUB-TOTAL
     BY IN
     BY NME AS NAME
     BY CDP
     BY TT AS 'TIME,TYPE'
IF ID IS &1
IF TT IS-NOT ' '
IF CDP IS-NOT ' '
IF SCW IS-NOT 0
END
FIN
END CATALOG
```

ROOT EXECUTIVE: R5

DATA BASE: Instructor

FUNCTION: Produce a standard report from data within the Instructor data base

The components of Executive Sequence R5 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	R5	EXEC
(2)	R5	RUNPROC

Listing of Filename = R5, Filetype = EXEC

```
&TYPE OFF
&STACK KT
&STACK EXEC R5
&STACK FIN
ATTACH RAMIS ALL BR
RAMIS
DETACH RAMIS BR
&STACK RT
&EXIT
```

Listing of Filename = R5, Filetype = RUNPROC

```
-DELETE
 FILE RAMEXEC
 1=OUT=R5,$
 END
 CATALOG R5
 DEFINE
 FILE INST
 IF ROT EQ 0 GOTO NONE ;
 LEFT/I4 = INT ( ROT / 100 ) ;
 IF LEFT GT 70 GOTO NOWYYMM ;
 PDATE/A5 = EDIT (ROT, '99/99') ;
 RIGHT/I4 = ROT - ( LEFT * 100 ) ;
 TEST/I6 = (RIGHT * 10000) + (LEFT * 100) + 30 ;
 GOTO GETDUR ;
 NOWYYMM ;
 PDATE/A5 = EDIT(ROT,'$$99')|'/'|EDIT(ROT,'99$$') ;
 TEST/I6 = (ROT * 100) + 30 ;
 GETDUR ;
 DURATION/I4 = YMD ( TEST, &YMD ) ;
 GOTO END ;
 NONE ;
 DURATION/I4 = 0 ;
 PDATE/A5 = '00/00' ;
 END
 TABLE
 HEADING
        *** DATE THIS REPORT : &DATE ***
 FILE INST
 COUNT ENTRIES
      BY SCH AS 'SCHOOL'
      BY PDATE AS 'PRD'
 IF DURATION IS-FROM 0 TO 180
 ON SCH SUB-TOTAL
 END
 END CATALOG
```

ROOT EXECUTIVE: R6

DATA BASE: Instructor

FUNCTION: Produce a standard report from data within the Instructor data base

The components of Executive Sequence R6 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype
(1)	R6	EXEC
(2)	R6	RUNPROC

Listing of Filename = R6, Filetype = EXEC

```
&TYPE OFF
&STACK KT
&STACK EXEC R6
&STACK FIN
ATTACH RAMIS ALL BR
RAMIS
DETACH RAMIS BR
&STACK RT
&EXIT
```

Listing of Filename = R6, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=R6,$
END
CATALOG R6
DEFINE
FILE INST
IF ROT EQ 0 GOTO NONE ;
LEFT/I4 = INT ( ROT / 100 ) ;
IF LEFT GT 70 GOTO NOWYYMM ;
PDATE/A5 = EDIT (ROT, '99/99') ;
RIGHT/I4 = ROT - ( LEFT * 100 ) ;
TEST/I6 = (RIGHT * 10000) + (LEFT * 100) + 30 ;
GOTO GETDUR ;
NOWYYMM ;
PDATE/A5 = EDIT(ROT,'$$99')|'/'|EDIT(ROT,'99$$') ;
TEST/I6 = (ROT * 100) + 30 ;
GETDUR ;
DURATION/I4 = YMD ( TEST, &YMD ) ;
GOTO END ;
NONE ;
DURATION/I4 = 0 ;
PDATE/A5 = '00/00' ;
END
TABLE
HEADING
       *** DATE THIS REPORT : &DATE ***
FILE INST
LIST ID AS SSN
AND NME AS NAME
AND IN
     BY SCH AS 'SCHOOL'
     BY PDATE AS 'PRD'
IF DURATION IS-FROM 0 TO 180
ON SCH SKIP-LINE
END
END CATALOG
```

ROOT EXECUTIVE: E40

DATA BASE: Course Master

FUNCTION: (1) Add a Course to an existing School, /or/
(2) Add a new School and Course

The sequence described below forms a CLASS A Prompting Type Executive. The file PD40/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD40/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E40 Logic

The components of Executive Sequence E40 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	E40	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE40	RAM	

Listing of Filename = E40, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD40 RE40
```

Listing of Filename = PD40, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER E40
1* FILE - COURSE
1* FUNCTION - ADD A NEW SCHOOL,
1*              AND/OR ADD A NEW COURSE
1*              TO AN EXISTING SCHOOL...
2SCH         11 4                    1                          1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP         21 4                    1                          5 4
3* ENTER COURSE CDP (CDP), A4
2DEL         31 1  1                 2                          9 1
3* ENTER DELETE FLAG (DEL), A1
3*    1 = COURSE ACTIVE
3*    2 = COURSE HAS BEEN DELETED
2CTLG        4110                    2                         1010
3* ENTER COURSE CATALOG NUMBER, A4
2CRS         5116                    2                         2016
3* ENTER COURSE TITLE (CRS), A16
2CONV1       62     0.0              240.0                      36 3
3* ENTER CONVENINGS NUMBER1 (CONV1), I3
2LEN1        73     0.0              240.0                    4 39 4
3* ENTER COURSE LENGTH #1 (LEN1), F4.1
2CAP1        82     0.0              240.0                      43 3
3* ENTER COURSE CAPACITY #1 (CAP1), I3
2BCAP1       92     0.0              240.0                      46 3
3* ENTER BUPERS CAPACITY #1 (BCAP1), I3
2DMD1       102     0.0              240.0                      49 5
3* ENTER COURSE DEMAND #1 (DMD1), I5
2BDMD1      112     0.0              240.0                      54 5
3* ENTER BUPERS DEMAND #1 (BDMD1), I5
2OFST       112     0.0              240.0                      59 2
3* ENTER SCHEDULE OFFSET (OFST), I2
2PFAIL      133     0.0              240.0                    5 61 5
3* ENTER % FAILURE RATE (PFAIL), F5.3
2NADIS      143     0.0              240.0                    5 66 5
3* ENTER % NONAC DISENROLL (NADIS), F5.3
2NOSHOW     153     0.0              240.0                    5 71 5
3* ENTER % NO-SHOW (NOSHOW), F5.3
2SETB       163     0.0              240.0                    5 76 5
3* ENTER % SETBACKS (SETB), F5.3
2BKLG       172     0.0              240.0                      81 4
3* ENTER NUMBER OF BACKLOGS (BKLG), I4
2JNO        181 4                                              85 4
3* COURSE J-NUMBER (JNO), A4
2CONV2      192     0.0              240.0                      89 3
3* ENTER CONVENINGS NUMBER2 (CONV2), I3
2LEN2       203     0.0              240.0                    4 92 4
3* ENTER COURSE LENGTH #2 (LEN2), F4.1
2CAP2       212     0.0              240.0                      96 3
3* ENTER COURSE CAPACITY #2 (CAP2), I3
2BCAP2      222     0.0              240.0                      99 3
3* ENTER BUPERS CAPACITY #2 (BCAP2), I3
2DMD2       232     0.0              240.0                     102 5
3* ENTER COURSE DEMAND #2 (DMD2), I5
2BDMD2      242     0.0              240.0                     107 5
3* ENTER BUPERS DEMAND #2 (BDMD2), I5
```

Listing of Filename = PD40, Filetype = DATA (Continued)

```
2CHWK2      252     0.0                 240.0                    112 3
3* ENTER CHANGE WEEK #2 (CHWK2), I3
2CONV3      262     0.0                 240.0                    115 3
3* ENTER CONVENINGS NUMBER3 (CONV3), I3
2LEN3       272     0.0                 240.0                    118 4
3* ENTER COURSE LENGTH #3 (LEN3), F4.1
2CAP3       282     0.0                 240.0                    122 3
3* ENTER COURSE CAPACITY #3 (CAP3), I3
2BCAP3      292     0.0                 240.0                    125 3
3* ENTER BUPERS CAPACITY #3 (BCAP3), I3
2DMD3       302     0.0                 240.0                    128 5
3* ENTER COURSE DEMAND #3 (DMD3), I5
2BDMD3      312     0.0                 240.0                    133 5
3* ENTER BUPERS DEMAND #3 (BDMD3), I5
2CHWK3      322     0.0                 240.0                    138 3
3* ENTER CHANGE WEEK #3 (CHWK3), I3
2PAOB       332     0.0                 240.0                    141 4
3* ENTER PLANNED AOB (PAOB), I4
2AFAIL      343     0.0                 240.0                    145 5
3* ENTER % ACAD FAILURES (AFAIL), F5.3
2PRTY       352     0.0                 240.0                    150 2
3* ENTER COURSE PRIORITY (PRTY), I2
2CTYPE      361 2                       2                        152 2
3* ENTER COURSE TYPE (CTYPE), A2
2PNEC       371 4                       2                        154 4
3* ENTER PRIMARY NEC (PNEC), A4
2UCAP       382     0.0                 240.0                    158 3
3* ENTER % CAPACITY UTILIZED (UCAP), I3
2UDEM       392     0.0                 240.0                    161 3
3* ENTER % DEMAND UTILIZED (UDEM), I3
2FPRTY      402     0.0                 240.0                    164 2
3* ENTER FTC PRIORITY (FPRTY), I2
2RDTE       412     0.0                 240.0                    166 6
3* ENTER REVISION DATE (RDTE),
2HSETB      423     0.0                 240.0                   5171 5
3* ENTER HIST. SETBACK (HSETB), F5.3
2TINSTR     433     0.0                 240.0                   4177 4
3* ENTER TOT INSTRS REQD (TINSTR), F4.1
2SUIC       441 5                       2                        181 5
3* ENTER STUDENT UIC (SUIC), A5
2CCMM       45110                       2                       18610
3* ENTER CCMM, A10
```

Listing of Filename = ESUB1, Filetype =EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE40, Filetype =RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,1,10,16,3,4,3,3,5,5,2,5,5,5,5,4,4,3,4,3,3,5,5,3,3
FORM 4,3,3,5,5,3,4,5,2,2,4,3,3,2,6,5,4,5,10
KEY 1,2
INCLUDE 1
FILE COURSE
END
```

ROOT EXECUTIVE: E41

DATA BASE: Course Master

FUNCTION: Add a new Student Instructor Ratio (SIR) and a new Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file PD41/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD41/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E41 Logic

The components of Executive Sequence E41 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	E41	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE41	RAM	

Listing of Filename = E41, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD41 RE41
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE41, Filetype - RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,X187,4,5,1,9,3,1
ORDER 1,2,46,47,48,49,50,51
KEY 1,2,46
INCLUDE 46
FILE COURSE
END
```

Listing of Filename = PD41, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER E41
1* FILE - COURSE
1* FUNCTION - ADD A STUDENT/INSTR RATIO
1*            AND AN INSTRUCTOR TO AN
1*            EXISTING COURSE...
2SCH        11 4                    1                    1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP        21 4                    1                    5 4
3* ENTER COURSE CDP (CDP), A4
2SIR       463    0.0               140.0              4196 4
3* ENTER STUDENT/INSTR RATIO (SIR), F4.1
2CHRS      473    0.0               240.0              2200 5
3* ENTER INSTR CONTACT HRS (CHRS), F5.1
2CTYP      481 1                    1                   205 1
3* ENTER COURSE TYPE (CTYP), A1
2ID        491 9                    1                   206 9
3* ENTER INSTRUCTOR ID (ID), A9
2QUAL      501 3                    2                   215 3
3* ENTER INSTR % QUALIFIED (QUAL), A3
2ASN       512 1  0.0               140.0               218 1
3* ENTER ASSIGNED FLAG (ASN), I1
```

ROOT EXECUTIVE: E42

DATA BASE: Course Master

FUNCTION: Add a new Instructor to an existing Student Instructor Ratio (SIR)

The sequence described below forms a CLASS A Prompting Type Executive. The file PD42/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD42/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E42 Logic

The components of Executive Sequence E42 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	E42	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE42	RAM	

Listing of Filename = E42, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD42 RE42
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE42, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,X187,4,X6,9,3,1
ORDER 1,2,46,49,50,51
KEY 1,2,46,49
INCLUDE 49
FILE COURSE
END
```

Listing of Filename = PD42, Filetype = DATA

```
2SCH          11 4                    1                          1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP          21 4                    1                          5 4
3* ENTER COURSE CDP (CDP), A4
2SIR         463    0.0               140.0                   4196 4
3* ENTER STUDENT/INSTR RATIO (SIR), F4.1
2ID          491 9                    1                        206 9
3* ENTER INSTRUCTOR ID (ID), A9
2QUAL        501 3                    2                        215 3
3* ENTER INSTR % QUALIFIED (QUAL), A3
2ASN         512 1  0.0               140.0                    218 1
3* ENTER ASSIGNED FLAG (ASN), I1
```

ROOT EXECUTIVE: E61

DATA BASE: Course Master

Function: Change an existing Field

This single executive serves to force entry to the SCAN mode of the RAMIS data management system. After SCAN mode has been successfully entered, the User may direct change of existing field(s) interactively through his terminal.

Terminal entry requirements of the RAMIS SCAN mode are fully covered in the NCSS vendor's documentation.

The DOTS User's Manual lists the data field names and identification numbers as appropriate for the Course Master data base.
This list will not be repeated in this document.

Listing of Executive E61 (SCAN Mode Entry)

```
&TYPE OFF
&STACK KT
&STACK SCAN
&STACK FILE COURSE
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
&EXIT
```

ROOT EXECUTIVE: E62

DATA BASE: Course Master

FUNCTION: Delete a School

The components of Executive Sequence E62 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	E62	EXEC
(2)	X62	RUNPROC

Listing of Filename = E62, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE E62
&PRI * FILE - COURSE
&PRI * FUNCTION - DELETE AN ENTIRE SCHOOL
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -OK
&PRI * WRONG NUMBER OF ARGUMENTS - RETRY
&GOTO -LP1
-OK &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC X62 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = X62, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X62,$
END
CATALOG X62
DELETE
FORM 4
SET 1
FILE COURSE
&1
END
END CATALOG
```

ROOT EXECUTIVE: E63

DATA BASE: Course Master

FUNCTION: Delete a Course

The components of Executive Sequence E63 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	E63	EXEC
(2)	X63	RUNPROC

Listing of Filename = E63, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER E63
&PRI * FILE - COURSE
&PRI * FUNCTION - DELETE A COURSE
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK &IF &1 EQ OUT &GOTO -OUT
&SAV1 = &1
-LP2 &PRI *
&PRI * ENTER COURSE CDP (CDP), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -OK
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP2
-OK &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC X63 &SAV1 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = X63, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X63,$
END
CATALOG X63
DELETE
FORM 4,X4,4
SET 2
FILE COURSE
&1
END
END CATALOG
```

ROOT EXECUTIVE: E64

DATA BASE: Course Master

FUNCTION: Delete an existing Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file PD64/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file PD64/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence E64 Logic

The components of Executive Sequence E64 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E64	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RE64	RAM	

Listing of Filename = E64, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 PD64 RE64
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RE64, Filetype = RAM

```
DELETE
READ TRANS FILE
RECORD 250
FORM 4,4,X187,4,X6,9
ORDER 1,2,46,49
SET 49
FILE COURSE
END
```

Listing of Filename = PD64, Filetype = DATA

```
2SCH        11 4                    1                          1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP        21 4                    1                          5 4
3* ENTER COURSE CDP (CDP), A4
2SIR       463    0.0               140.0                    4196 4
3* ENTER STUDENT/INSTR RATIO (SIR), F4.1
2ID        491 9                    1                        206 9
3* ENTER INSTRUCTOR ID (ID), A9
```

ROOT EXECUTIVE: E65

DATA BASE: Course Master

FUNCTION: Delete an Instructor

The components of Executive Sequence E65 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	E65	EXEC
(2)	X65	RUNPROC

Listing of Filename = E65, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER E65
&PRI * FILE - COURSE
&PRI * FUNCTION - DELETE A STUDENT/INSTRUCTOR RATIO FROM A COURSE
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &GOTO -OUT
&SAV1 = &1
-LP2 &PRI *
&PRI * ENTER COURSE CDP (CDP), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK2
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP2
-CK2 &IF &1 EQ OUT &GOTO -OUT
&SAV2 = &1
-LP3 &PRI *
&PRI * ENTER STUDENT/INSTRUCTOR RATIO (SIR), F4.1
&READ ARGS
&IF &INDEX EQ 1 &GOTO -OK
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP3
-OK &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC X65 &SAV1 &SAV2 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = X65, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X65,$
END
CATALOG X65
DELETE
FORM 4,X4,4,X4,4
ORDER 1,2,46
SET 46
FILE COURSE
&1
END
END CATALOG
```

ROOT EXECUTIVE: E43

DATA BASE: Course Master

FUNCTION: Load the Course Master data base From card input

The Course Master data base is loaded from scratch via the following Executive Sequences:

Executive Sequence	Input Card Format	Input Data Contents
E43	Course File, TYPE -1- and TYPE -2-	Fixed data relative to individual Courses
E44	Course File, TYPE -3-	Extension of individual Course data
E34	Instructor File, TYPE -2-	Assignment of individual Instructors to relevant Courses

In the creation of a new Course Master data set, the Executive Sequences should be executed in the order listed above.

If no TYPE -3- Course file data cards are available, they may be omitted from the load sequence and Executive Sequence E44 may then be run at the convenience of the User.

If Executive Sequences E43 and E34 are to be run, the order must be E43; then, E34. E43 must run successfully, if error-free operation of the E34 Sequence is anticipated.

The format of the card input data sets associated with Executive Sequence E43 is illustrated in the diagram following entitled "Course Data Base Type -1- Input Card Format", and "Course Data Base Type -2- Input Card Format". These card formats must be merged with the Type -1- and Type -2- cards for each course placed adjacent in the input; a Type -1- card precedes the related Type -2- card for the same Course.

Executive Sequence E43 Logic

The components of Executive Sequence E43 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	E43	EXEC	
(2)	PGM15	PLI	(PL/1 Program)
(3)	RE43	RAM	

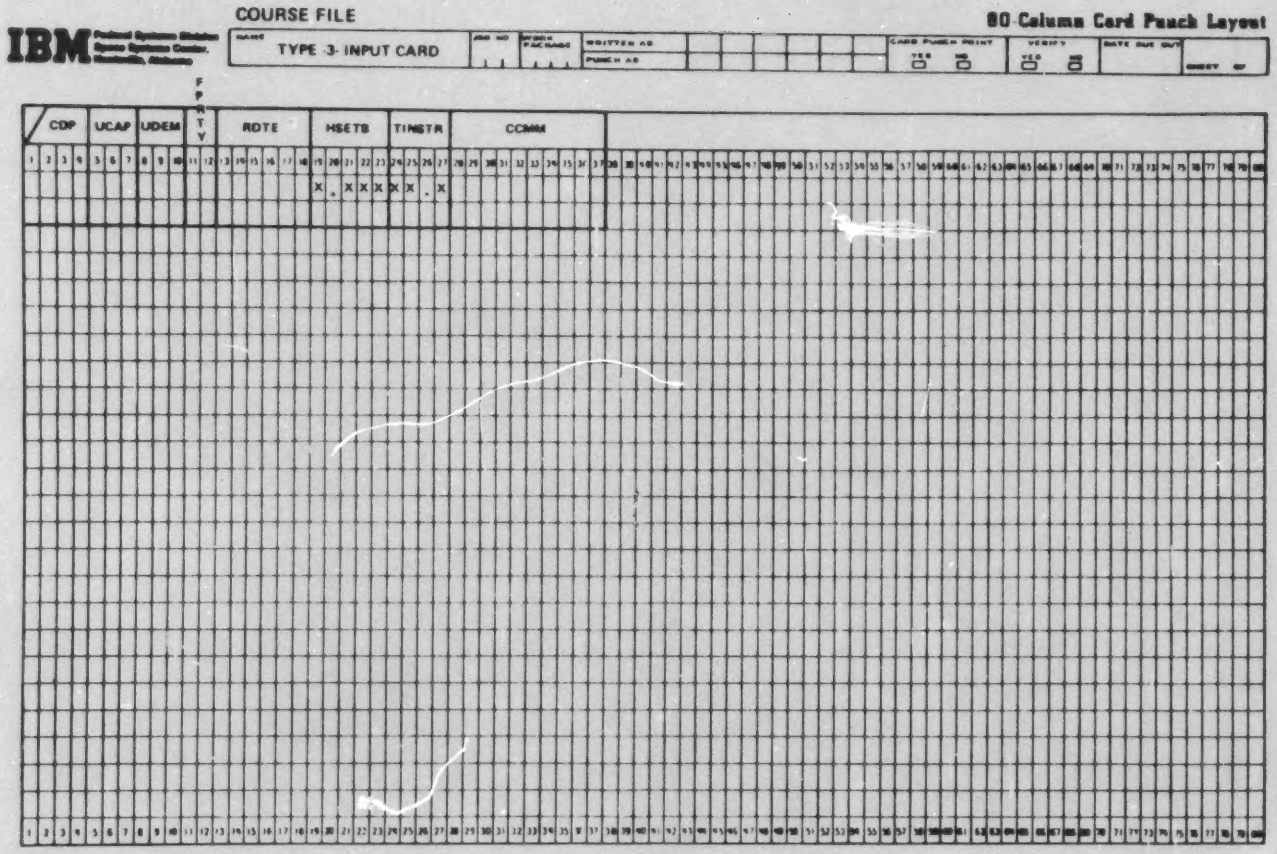
Listing of Filename = E43, Filetype = EXEC

```
&TYPE OFF
STATE E43 DATA
&IF &INDEX0 NE 0 &GOTO -NODATA
FILEDEF CARDS DSK E43 DATA
FILEDEF TRANS DSK COURSE INPUT LR 250 RE F
LOAD PGM16 (CLEAR LIBE) PLILIB
START (BRIEF)
FILEDEF CARDS CLEAR
FILEDEF TRANS CLEAR
ERASE E43 DATA
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE43 RAM
DETACH RAMIS BR
&EXIT
-NODATA &PRINT NO COURSE CARDS PROVIDED -- FUNCTION TERMINATING
```

Listing of Filename = RE43, Filetype = RAM

```
ERASE
FILE COURSE
END
INPUT
READ COURSE INPUT
RECORD 250
SET 1
FORM 4,4,1,10,16,3,4,3,3,5,5,2,5,5,5,5,4,4,3,4,3,3,5,5,3,3,4,3,3,5,5,3
FORM 4,5,2,2,4,3,3,2,6,5,4,5,10,4,5,1,9,3,1
FILE COURSE
END
```

COURSE DATA BASE TYPE -1- INPUT CARD FORMAT



COURSE DATA BASE TYPE -2- INPUT CARD FORMAT

IBM Federal Systems Division Space Systems Center, Huntsville, Alabama

INSTRUCTOR FILE

80-Column Card Punch Layout

NAME: TYPE -2- CARD

WRITTEN AS

PUNCH AS

CARD PUNCH PRINT YES NO

VERIFY YES NO

DATE DUE OUT

SHEET OF

ID	NME	IN	SCH	CDP	TTL	CTYP	ASR	QUAL	SIR	CHRS
									X X . X	X X X . X

955-1115-0

ROOT EXECUTIVE: E44

DATA BASE: Course Master

FUNCTION: Load the Course Master data base From card input

The Course Master data base is loaded from scratch via the following Executive Sequences:

Executive Sequence	Input Card Format	Input Data Contents
E43	Course File, TYPE -1- and TYPE -2-	Fixed data relative to individual Courses
E44	Course File, TYPE -3-	Extension of individual Course data
E34	Instructor File, TYPE -2-	Assignment of individual Instructors to relevant Courses

In the creation of a new Course Master data set, the Executive Sequences should be executed in the order listed above.

If no TYPE -3- Course file data cards are available, they may be omitted from the load sequence and Executive Sequence E44 may then be run at the convenience of the User.

If Executive Sequences E43 and E34 are to be run, the order must be E43; then, E34. E43 must run successfully, if error-free operation of the E34 Sequence is anticipated.

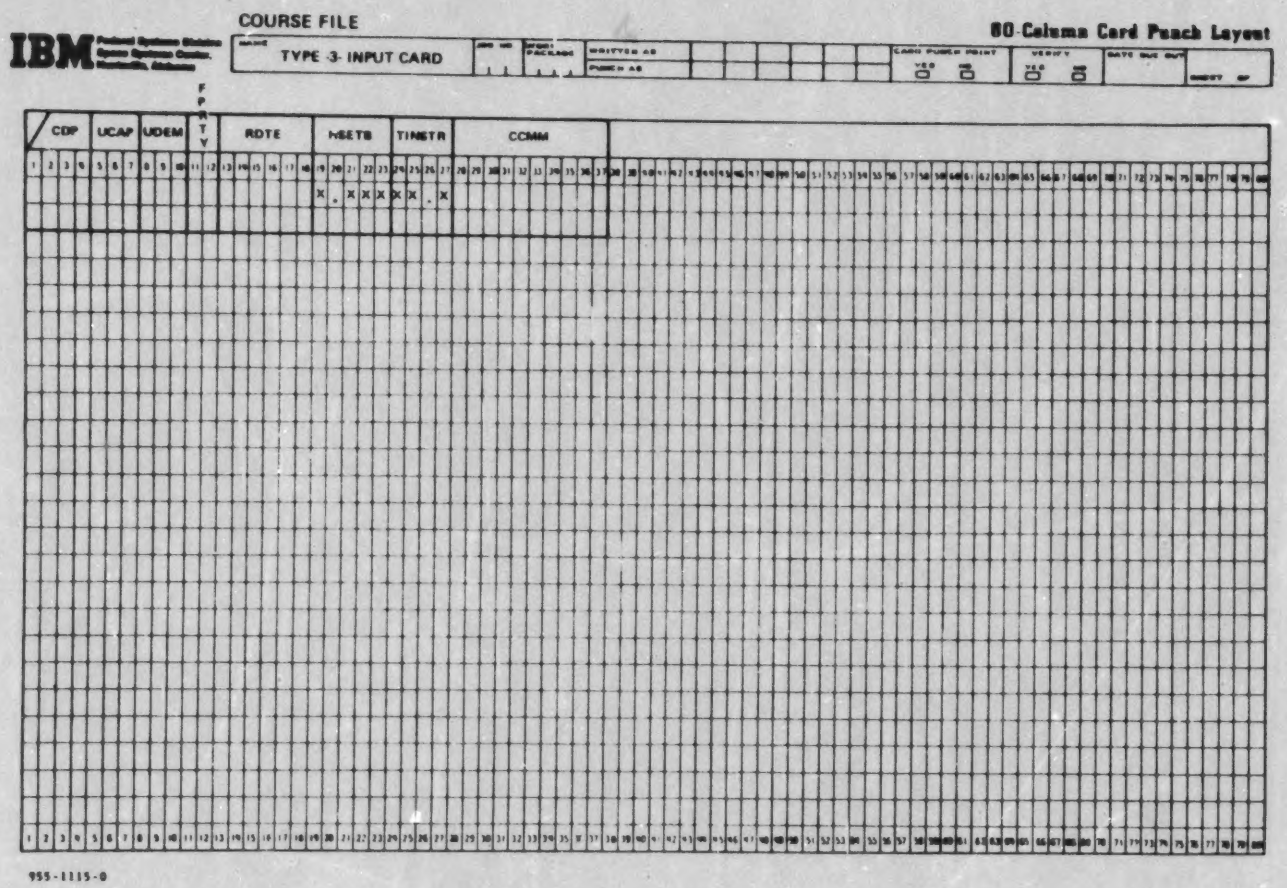
The format of the card input data set associated with Executive Sequence E44 is illustrated in the diagram following entitled "Course Data Base Type -3- Input Card Format".

Executive Sequence E44 Logic

The components of Executive Sequence E44 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype
(1)	E44	EXEC
(2)	RE44	RAM

COURSE DATA BASE TYPE -3- INPUT CARD FORMAT



Listing of Filename = E44, Filetype = EXEC

```
&TYPE OFF
STATE E44 DATA
&IF &INDEX0 NE 0 &GOTO -NODATA
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE44 RAM
DETACH RAMIS BR
ERASE E44 DATA
&EXIT
-NODATA &PRI * NO TYPE3 CARDS FOUND -- FUNCTION TERMINATING
```

Listing of Filename = RE44, Filetype = RAM

```
REVISE
READ E44 DATA
RECORD 80
FORM 4,3,3,2,6,5,4,10
ORDER 2,38,39,40,41,42,43,45
KEY 2
UPDATE 38
INCLUDE 53
FILE COURSE
END
```

ROOT EXECUTIVE: E34

DATA BASE: Course Master

FUNCTION: Load the Course Master data base from card input

The Course Master data base is loaded from scratch via the following Executive Sequences:

Executive Sequence	Input Card Format	Input Data Contents
E43	Course File, TYPE -1- and TYPE -2-	Fixed data relative to individual Courses
E44	Course File, TYPE -3-	Extension of individual Course data
E34	Instructor File, TYPE -2-	Assignment of individual Instructors to relevant Courses

In the creation of a new Course Master data set, the Executive Sequences should be executed in the order listed above.

If no TYPE -3- Course file data cards are available, they may be omitted from the load sequence and Executive Sequence E44 may then be run at the convenience of the User.

If Executive Sequences E43 and E34 are to be run, the order must be E43; then, E34. E43 must run successfully, if error-free operation of the E34 Sequence is anticipated.

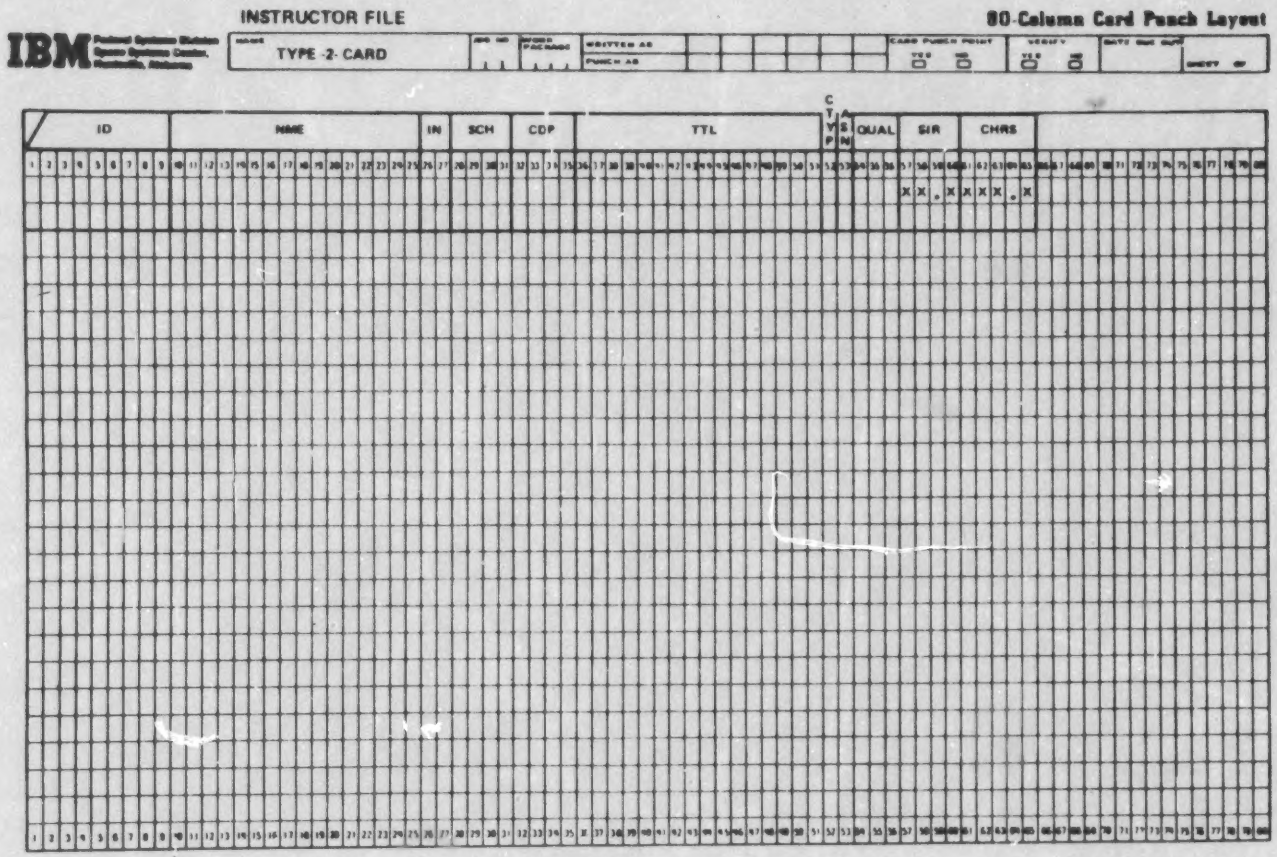
The format of the card input data set associated with Executive Sequence E34 is illustrated in the diagram following entitled "Instructor (Course) Data Base Type -2- Input Card Format".

Executive Sequence E34 Logic

The components of Executive Sequence E34 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype
(1)	E34	EXEC
(2)	RE34	RAM

INSTRUCTOR (COURSE) DATA BASE TYPE -2- INPUT CARD FORMAT



Listing of Filename = E34, Filetype = EXEC

```
&TYPE OFF
STATE E34 DATA
&IF &INDEX0 NE 0 &GOTO -NOCDS
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE34 RAM
DETACH RAMIS BR
ERASE E34 DATA
&EXIT
-NOCDS &PRINT NO TYPE2 CARDS FOUND -- FUNCTION TERMINATING
```

Listing of Filename = RE34, Filetype = RAM

```
REVISE
READ E34 DATA
RECORD 80
FORM 9,X18,4,4,X16,1,1,3,4,5
ORDER 49,1,2,48,51,50,46,47
KEY 1,2,46,49
INCLUDE 46
FILE COURSE
END
```

ROOT EXECUTIVE: E51

DATA BASE: Course Master
Course Scratch

FUNCTION: Recreate the Scratch data base using the records from one User-selected school in the Master data base

The first step in the recreation of the Scratch data base is the erase of the existing Scratch copy.

After being prompted, the User must supply the 4-character name of the school of interest.

The Master data base is then dumped into card-image format onto a temporary file, selecting only those records associated with the school chosen by the User. The temporary file is then used as input to recreate the Scratch data base in its entirety.

At the successful conclusion of this Executive Sequence the contents of the Master data base are unchanged. The new Scratch data base will contain only the records associated with the selected school.

Executive Sequence E51 Logic

The components of Executive Sequence E51 and the chronoligical order of execution are indicated in the table presented below. A listing of these files (except for programs) follows:

Sequence	Filename	Filetype	
(1)	E51	EXEC	
(2)	X68	RUNPROC	
(3)	PEM20	PLI	(PL/1 Program)
(4)	RE49	RAM	

Listing of Filename = E51, Filetype = EXEC

```
&TYPE OFF
ATTACH TEMP5 AS 193 BRIEF
&PRI * FILE - COURSE
&PRI * FUNCTION - SELECT SCHOOL FOR MODEL INPUT
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &EXIT
FILEDEF RAMUTIL1 DSK COURSE DUMP U LR 80 RE FB BL 800
&STACK KT
&STACK EXEC X68 &1
&STACK FIN
ATTACH RAMIS ALL BR
FILEDEF 6 MSG KILL
EXEC RAMIS
DETACH RAMIS BR
&STACK RT
FILEDEF RAMUTIL1 CLEAR
FILEDEF 6 CONO
&STACK KT
&STACK L /D=/
&STACK Q
EDIT MSG KILL
&PRI &1 SCHOOL DUMPED TO DISK FILE
FILEDEF TRANS DSK UREFOR COURSE U LR 218 RE F
FILEDEF CARDS DSK COURSE DUMP U LR 80 RE FB BL 800
LOAD PGM20 (CLEAR LIBE) PLILIB
START (BRIEF)
FILEDEF TRANS CLEAR
FILEDEF CARDS CLEAR
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE49 RAM
DETACH RAMIS BR
DETACH 193 BR
&EXIT
```

Listing of Filename = X68, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=X68,$
END
CATALOG X68
DUMP
FILE COURSE
PRINT ASN AND SAVE
IF SCH IS &1
END
END CATALOG
```

Listing of Filename = RE49, Filetype = RAM

```
ERASE
FILE MODCOURSE
END
INPUT
READ UREFOR COURSE
RECORD 218
FORM 4,4,1,10,16,3,4,3,3,5,5,2,5,5,5,5,4,4,3,4,3,3,5,5,3,3,4,3,3,5,5
FORM 4,5,2,2,4,3,3,2,6,5,4,5,10,4,5,1,9,3,1
SET 1
SEARCH FORWARD
FILE MODCOURSE
END
```

ROOT EXECUTIVE: E50

DATA BASE: Course Master
Course Scratch

FUNCTION: Recreate the Scratch data base using the records from all Schools in the Master data base

The first step in the recreation of the Scratch data base is the erase of the existing Scratch copy.

The Master is then dumped into card-image format onto a temporary dataset. This file is then used as input to recreate the Scratch data base in its entirety.

Executive Sequence E50 Logic

The components of Executive Sequence E50 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	E50	EXEC	
(2)	RE45	RAM	
(3)	PGM20	PLI	(PL/1 Program)
(4)	RE49	RAM	

Listing of Filename = E50, Filetype = EXEC

```
&TYPE OFF
ATTACH TEMP5 AS 193 BRIEF
FILEDEF RAMUTIL1 DSK COURSE DUMP U LR 80 RE FB BL 800
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE45 RAM
DETACH RAMIS BR
FILEDEF RAMUTIL1 CLEAR
FILEDEF TRANS DSK UREFOR COURSE U LR 218 RE F
FILEDEF CARDS DSK COURSE DUMP U LR 80 RE FB BL 800
LOAD PGM20 (CLEAR LIBE) PLILIB
START (BRIEF)
FILEDEF TRANS CLEAR
FILEDEF CARDS CLEAR
&STACK KT
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMISIN RE49 RAM
DETACH 193 BRIEF
DETACH RAMIS BR
&EXIT
```

Listing of Filename = RE45, Filetype = RAM

```
DUMP
FILE COURSE
PRINT ASN AND SAVE
END
```

Listing of Filename = RE49, Filetype = RAM

```
ERASE
FILE MODCOURSE
END
INPUT
READ UREFOR COURSE
RECORD 218
FORM 4,4,1,10,16,3,4,3,3,5,5,2,5,5,5,5,4,4,3,4,3,3,5,5,3,3,4,3,3,5,5,3
FORM 4,5,2,2,4,3,3,2,6,5,4,5,10,4,5,1,9,3,1
SET 1
SEARCH FORWARD
FILE MODCOURSE
END
```

ROOT EXECUTIVE: E58

DATA BASE: Course Master

FUNCTION: Dump all records to the User's terminal

This Executive Sequence produces a formatted dump to the User's terminal of all of the records of the Course Master data base.

Executive Sequence E58 Logic

The components of Executive Sequence E58 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	E58	EXEC	
(2)	RE45	RAM	
(3)	PGM20	PLI	(PL/1 Program)

Listing of Filename = E58, Filetype = EXEC

```
&TYPE OFF
ATTACH TEMP5 AS 193 BRIEF
FILEDEF RAMUTIL1 DSK COURSE DUMP U LR 80 RE FB BL 800
&STACK KT
ATTACH RAMIS ALL BR
EXEC RAMISIN RE45 RAM
DETACH RAMIS BR
FILEDEF RAMUTIL1 CLEAR
FILEDEF DUMP DSK COURSE DUMP U LR 80 RE FB BL 800
&STACK RT
LOAD PGM19 (CLEAR LIBE) PLILIB
START (BRIEF)
DETACH 193 BRIEF
FILEDEF DUMP CLEAR
```

Listing of Filename = RE45, Filetype = RAM

```
DUMP
FILE COURSE
PRINT ASN AND SAVE
END
```

ROOT EXECUTIVE: E37

DATA BASE: Course Scratch

FUNCTION: Dump all records to the User's terminal

This Executive Sequence produces a formatted dump to the User's terminal of all of the records of the Course Scratch data base.

Executive Sequence E37 Logic

The components of Executive Sequence E37 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	E37	EXEC	
(2)	RE37	RAM	
(3)	PGM19	PLI	(PL/1 Program)

Listing of Filename = E37, Filetype = EXEC

```
&TYPE OFF
ATTACH TEMP5 AS 193 BR
FILEDEF RAMUTIL1 DSK MCOURSE DUMP U LR 80 RE FB BL 800
&STACK KT
ATTACH RAMIS ALL BR
EXEC RAMISIN RE37 RAM
DETACH RAMIS BR
FILEDEF RAMUTIL1 CLEAR
FILEDEF DUMP DSK MCOURSE DUMP U LR 80 RE FB BL 800
&STACK RT
LOAD PGM19 (CLEAR LIBE) PLILIB
START (BRIEF)
DETACH 193 BR
FILEDEF DUMP CLEAR
```

Listing of Filename = RE37, Filetype = RAM

```
DUMP
FILE MODCOURSE
PRINT ASN AND SAVE
END
```

ROOT EXECUTIVE: W40

DATA BASE: Course Scratch

FUNCTION: (1) Add a Course to an existing School, /or/
(2) Add a new School and Course

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD40/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD40/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W40 Logic

The components of Executive Sequence W40 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	W40	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW40	RAM	

Listing of Filename = W40, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD40 RW40
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW40, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,1,10,16,3,4,3,3,5,5,2,5,5,5,5,4,4,3,4,3,3,5,5,3,3
FORM 4,3,3,5,5,3,4,5,2,2,4,3,3,2,6,5,4,5,10
KEY 1,2
INCLUDE 1
FILE MODCOURSE
END
```

Listing of Filename = WPD40, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER W40
1* FILE - COURSE SCRATCH DATA BASE
1* FUNCTION - ADD A NEW SCHOOL,
1*              AND/OR ADD A NEW COURSE
1*              TO AN EXISTING SCHOOL...
2SCH        11 4                    1                    1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP        21 4                    1                    5 4
3* ENTER COURSE CDP (CDP), A4
2DEL        31 1  1                 2                    9 1
3* ENTER DELETE FLAG (DEL), A1
3*    1 = COURSE ACTIVE
3*    2 = COURSE HAS BEEN DELETED
2CTLG       4110                    2                   1010
3* ENTER COURSE CATALOG NUMBER, A4
2CRS        5116                    2                   2016
3* ENTER COURSE TITLE (CRS), A16
2CONV1      62    0.0               240.0                36 3
3* ENTER CONVENINGS NUMBER1 (CONV1), I3
2LEN1       73    0.0               240.0              4 39 4
3* ENTER COURSE LENGTH #1 (LEN1), F4.1
2CAP1       82    0.0               240.0                43 3
3* ENTER COURSE CAPACITY #1 (CAP1), I3
2BCAP1      92    0.0               240.0                46 3
3* ENTER BUPERS CAPACITY #1 (BCAP1), I3
2DMD1      102    0.0               240.0                49 5
3* ENTER COURSE DEMAND #1 (DMD1), I5
2BDMD1     112    0.0               240.0                54 5
3* ENTER BUPERS DEMAND #1 (BDMD1), I5
2OFST      112    0.0               240.0                59 2
3* ENTER SCHEDULE OFFSET (OFST), I2
2PFAIL     133    0.0               240.0              5 61 5
3* ENTER % FAILURE RATE (PFAIL), F5.3
2NADIS     143    0.0               240.0              5 66 5
3* ENTER % NONAC DISENROLL (NADIS), F5.3
2NOSHOW    153    0.0               240.0              5 71 5
3* ENTER % NO-SHOW (NOSHOW), F5.3
2SETB      163    0.0               240.0              5 76 5
3* ENTER % SETBACKS (SETB), F5.3
2BKLG      172    0.0               240.0                81 4
3* ENTER NUMBER OF BACKLOGS (BKLG), I4
2JNO       181 4                                         85 4
3* COURSE J-NUMBER (JNO), A4
2CONV2     192    0.0               240.0                89 3
3* ENTER CONVENINGS NUMBER2 (CONV2), I3
2LEN2      203    0.0               240.0              4 92 4
3* ENTER COURSE LENGTH #2 (LEN2), F4.1
2CAP2      212    0.0               240.0                96 3
3* ENTER COURSE CAPACITY #2 (CAP2), I3
2BCAP2     222    0.0               240.0                99 3
3* ENTER BUPERS CAPACITY #2 (BCAP2), I3
2DMD2      232    0.0               240.0               102 5
3* ENTER COURSE DEMAND #2 (DMD2), I5
2BDMD2     242    0.0               240.0               107 5
3* ENTER BUPERS DEMAND #2 (BDMD2), I5
```

Listing of Filename = WPD40, Filetype = DATA (Continued)

```
2CHWK2      252    0.0              240.0                    112 3
3* ENTER CHANGE WEEK #2 (CHWK2), I3
2CONV3      262    0.0              240.0                    115 3
3* ENTER CONVENINGS NUMBER3 (CONV3), I3
2LEN3       273    0.0              240.0                    118 4
3* ENTER COURSE LENGTH #3 (LEN3), F4.1
2CAP3       282    0.0              240.0                    122 3
3* ENTER COURSE CAPACITY #3 (CAP3), I3
2BCAP3      292    0.0              240.0                    125 3
3* ENTER BUPERS CAPACITY #3 (BCAP3), I3
2DMD3       302    0.0              240.0                    128 5
3* ENTER COURSE DEMAND #3 (DMD3), I5
2BDMD3      312    0.0              240.0                    133 5
3* ENTER BUPERS DEMAND #3 (BDMD3), I5
2CHWK3      322    0.0              240.0                    138 3
3* ENTER CHANGE WEEK #3 (CHWK3), I3
2PAOB       332    0.0              240.0                    141 4
3* ENTER PLANNED AOB (PAOB), I4
2AFAIL      343    0.0              240.0                    145 5
3* ENTER % ACAD FAILURES (AFAIL), F5.3
2PRTY       352    0.0              240.0                    150 2
3* ENTER COURSE PR.ORITY (PRTY), I2
2CTYPE      361 2                   2                        152 2
3* ENTER COURSE TYPE (CTYPE), A2
2PNEC       371 4                   2                        154 4
3* ENTER PRIMARY NEC (PNEC), A4
2UCAP       382    0.0              240.0                    158 3
3* ENTER % CAPACITY UTILIZED (UCAP), I3
2UDEM       392    0.0              240.0                    161 3
3* ENTER % DEMAND UTILIZED (UDEM), I3
2FPRTY      402    0.0              240.0                    164 2
3* ENTER FTC PRIORITY (FPRTY), I2
2RDTE       412    0.0              240.0                    166 6
3* ENTER REVISION DATE (RDTE),
2HSETB      423    0.0              240.0                   5171 5
3* ENTER HIST. SETBACK (HSETB), F5.3
2TINSTR     433    0.0              240.0                   4177 4
3* ENTER TOT INSTRS REQD (TINSTR), F4.1
2SUIC       441 5                   2                        181 5
3* ENTER STUDENT UIC (SUIC), A5
2CCMM       45110                   2                        18610
3* ENTER CCMM, A10
```

ROOT EXECUTIVE: W41

DATA BASE: Course Scratch

FUNCTION: Add a new Student Instructor Ratio (SIR) and a new Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD41/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD41/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W41 Logic

The components of Executive Sequence W41 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	W41	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW41	EXEC	

Listing of Filename = W41, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD41 RW41
```

Listing of Filename = ESUB1, Filetype =EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW41, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,X187,4,5,1,9,3,1
ORDER 1,2,46,47,48,49,50,51
KEY 1,2,46
INCLUDE 46
FILE MODCOURSE
END
```

Listing of Filename = WPD41, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER W41
1* FILE - COURSE SCRATCH DATA BASE
1* FUNCTION - ADD A STUDENT/INSTR RATIO
1*           AND AN INSTRUCTOR TO AN
1*           EXISTING COURSE...
2SCH        11 4                    1                    1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP        21 4                    1                    5 4
3* ENTER COURSE CDP (CDP), A4
2SIR       463    0.0               140.0              4196 4
3* ENTER STUDENT/INSTR RATIO (SIR), F4.1
2CHRS      473    0.0               240.0              2200 5
3* ENTER INSTR CONTACT HRS (CHRS), F5.1
2CTYP      481 1                    1                   205 1
3* ENTER COURSE TYPE (CTYP), A1
2ID        491 9                    1                   206 9
3* ENTER INSTRUCTOR ID (ID), A9
2QUAL      501 3                    2                   215 3
3* ENTER INSTR % QUALIFIED (QUAL), A3
2ASN       512 1  0.0               140.0               218 1
3* ENTER ASSIGNED FLAG (ASN), I1
```

ROOT EXECUTIVE: W42

DATA BASE: Course Scratch

FUNCTION: Add a new Instructor to an existing Student Instructor Ratio (SIR)

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD42/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD42/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W42 Logic

The components of Executive Sequence W42 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	W42	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW42	RAM	

Listing of Filename = W42, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD42 RW42
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW42, Filetype = RAM

```
REVISE
READ TRANS FILE
RECORD 250
FORM 4,4,X187,4,X6,9,3,1
ORDER 1,2,46,49,50,51
KEY 1,2,46,49
INCLUDE 49
FILE MODCOURSE
END
```

Listing of Filename = WPD42, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER W42
1* FILE - COURSE SCRATCH DATA BASE
1* FUNCTION - ADD A NEW INSTRUCTOR TO AN
1*              EXISTING STUDENT/INSTR
1*              RATIO AND COURSE...
2SCH         11 4                    1                          1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP         21 4                    1                          5 4
3* ENTER COURSE CDP (CDP), A4
2SIR        463    0.0               140.0                    4196 4
3* ENTER STUDENT/INSTR RATIO (SIR), F4.1
2CHRS       473    0.0               240.0                    2200 5
3* ENTER INSTR CONTACT HRS (CHRS), F5.1
2CTYP       481 1                    1                         205 1
3* ENTER COURSE TYPE (CTYP), A1
2ID         491 9                    1                         206 9
3* ENTER INSTRUCTOR ID (ID), A9
2QUAL       501 3                    2                         215 3
3* ENTER INSTR % QUALIFIED (QUAL), A3
2ASN        512 1  0.0               140.0                     218 1
3* ENTER ASSIGNED FLAG (ASN), I1
```

ROOT EXECUTIVE: W61

DATA BASE: Course Scratch

FUNCTION: Change an existing Field

This single executive serves to force entry to the SCAN mode of the RAMIS data management system. After SCAN mode has been successfully entered, the user may direct change of existing field(s) interactively through his terminal.

Terminal entry requirements of the RAMIS SCAN mode are fully covered in the NCSS vendor's documentation.

The DOTS User's Manual lists the data field names and identification numbers as appropriate for the Course Scratch data base.
This list will not be repeated in this document.

Listing of Executive W61 (SCAN Mode Entry)

```
&TYPE OFF
&BEGSTACK
KT
SCAN
FILE MODCOURSE
&ENDSTACK
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
&EXIT
```

ROOT EXECUTIVE: W62

DATA BASE: Course Scratch

FUNCTION: Delete a School

The components of Executive Sequence W62 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	W62	EXEC
(2)	WX62	RUNPROC

Listing of Filename = W62, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE W62
&PRI * FILE - MODCOURSE
&PRI * FUNCTION - DELETE AN ENTIRE SCHOOL
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -OK
&PRI * WRONG NUMBER OF ARGUMENTS - RETRY
&GOTO -LP1
-OK &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC WX62 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = WX62, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=WX62,$
END
CATALOG WX62
DELETE
FORM 4
SET 1
FILE MODCOURSE
&1
END
END CATALOG
```

ROOT EXECUTIVE: W63

DATA BASE: Course Scratch

FUNCTION: Delete a Course

The components of Executive Sequence W63 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	W63	EXEC
(2)	WX63	RUNPROC

Listing of Filename = W63, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER W63
&PRI * FILE - MODCOURSE
&PRI * FUNCTION - DELETE A COURSE
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK &IF &1 EQ OUT &GOTO -OUT
&SAV1 = &1
-LP2 &PRI *
&PRI * ENTER COURSE CDP (CDP), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -OK
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP2
-OK &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC WX63 &SAV1 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = WX63, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=WX63,$
END
CATALOG WX63
DELETE
FORM 4,X4,4
SET 2
FILE MODCOURSE
&1
END
END CATALOG
```

ROOT EXECUTIVE: W64

DATA BASE: Course Scratch

FUNCTION: Delete an existing Instructor

The sequence described below forms a CLASS A Prompting Type Executive. The file WPD64/DATA contains both the texual character strings to be used in prompting the User and all necessary formatting and editing criteria as required to process the entries made at the terminal.

The record format of the file WPD64/DATA is described fully in the documentation associated with the PL/1 program called 'PROMPT' and is not duplicated herein; a listing of the contents of the file is included in this section.

Executive Sequence W64 Logic

The components of Executive Sequence W64 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

Sequence	Filename	Filetype	
(1)	W64	EXEC	
(2)	ESUB1	EXEC	
(3)	PROMPT	PLI	(PL/1 Program)
(4)	RW64	RAM	

Listing of Filename = W64, Filetype = EXEC

```
&TYPE OFF
EXEC ESUB1 WPD64 RW64
&EXIT
```

Listing of Filename = ESUB1, Filetype = EXEC

```
&TYPE OFF
FILEDEF TAB DSK &1 DATA LR 80 RE F
FILEDEF TRANS DSK TRANS FILE LR 250 RE F
LOAD PROMPT (CLEAR LIBE) PLILIB
START (BRIEF)
&IF &INDEX0 NE 0 &GOTO -END
&STACK KT
ATTACH RAMIS ALL BRIEF
FILEDEF 6 MSG KILL
EXEC RAMISIN &2 RAM
DETACH RAMIS BRIEF
EXEC RAMOUT
-END FILEDEF TAB CLEAR
FILEDEF TRANS CLEAR
&EXIT
```

Listing of Filename = RW64, Filetype = RAM

```
DELETE
READ TRANS FILE
RECORD 250
FORM 4,4,X187,4,X6,9
ORDER 1,2,46,49
SET 49
FILE MODCOURSE
END
```

Listing of Filename = WPD64, Filetype = DATA

```
1* THIS IS EXECUTIVE NUMBER W64
1* FILE - COURSE SCRATCH DATA BASE
1* FUNCTION - DELETE AN INSTRUCTOR FROM
1*            AN EXISTING STUDENT/INSTR
1*            RATIO AND COURSE...
2SCH        11 4                  1                    1 4
3* ENTER SCHOOL CODE (SCH), A4
2CDP        21 4                  1                    5 4
3* ENTER COURSE CDP (CDP), A4
2SIR       463    0.0             140.0              4196 4
3* ENTER STUDENT/INSTR RATIO (SIR), F4.1
2ID        491 9                  1                  206 9
3* ENTER INSTRUCTOR ID (ID), A9
```

ROOT EXECUTIVE: W65

DATA BASE: Course Scratch

FUNCTION: Delete a Student/Instructor Ratio (SIR)

The components of Executive Sequence W65 and the chronological order of execution are indicated in the table presented below. A listing of these files (except for programs) follows.

This Executive Sequence is referred to in the DOTS User documentation as a CLASS B Prompting Type Executive.

Sequence	Filename	Filetype
(1)	W65	EXEC
(2)	WX65	RUNPROC

Listing of Filename = W65, Filetype = EXEC

```
&TYPE OFF
&PRI * THIS IS EXECUTIVE NUMBER W65
&PRI * FILE - MODCOURSE
&PRI * FUNCTION - DELETE A STUDENT/INSTRUCTOR RATIO FROM A COURSE
&PRI * ENTER OUT AT ANY TIME FOR PREMATURE EXIT
-LP1 &PRI *
&PRI * ENTER SCHOOL CODE (SCH), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK1
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP1
-CK1 &IF &1 EQ OUT &GOTO -OUT
&SAV1 = &1
-LP2 &PRI *
&PRI * ENTER COURSE CDP (CDP), A4
&READ ARGS
&IF &INDEX EQ 1 &GOTO -CK2
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP2
-CK2 &IF &1 EQ OUT &GOTO -OUT
&SAV2 = &1
-LP3 &PRI *
&PRI * ENTER STUDENT/INSTRUCTOR RATIO (SIR), F4.1
&READ ARGS
&IF &INDEX EQ 1 &GOTO -OK
&PRI * WRONG NUMBER OF ARGUMENTS -- RETRY
&GOTO -LP3
-OK &IF &1 EQ OUT &GOTO -OUT
&STACK KT
&STACK EXEC WX65 &SAV1 &SAV2 &1
&STACK FIN
ATTACH RAMIS ALL BR
&STACK RT
EXEC RAMIS
DETACH RAMIS BR
-OUT &EXIT
```

Listing of Filename = WX65, Filetype = RUNPROC

```
DELETE
FILE RAMEXEC
1=OUT=WX65,$
END
CATALOG WX65
DELETE
FORM 4,X4,4,X4,4
ORDER 1,2,46
SET 46
FILE MODCOURSE
&1
END
END CATALOG
```

PROGRAM NAME: PGM5

TYPE: PL/1

SYSTEM: Data Base Maintenance

NCSS FILENAME/FILETYPE: PGM5/PLI

A listing of the program source code and machine run flowcharts are included in this document on the pages following.

FUNCTION

This program converts the data found on the weekly Instructor Time Cards to a transaction format for entry to the Instructor Master data base. Output records (transactions) from this program are subsequently processed into the data base by the RAMIS data maintenance UPDATE function RE22/RAM (filename/filetype).

Any keypunch errors (invalid characters) found within the input record numeric fields will be diagnosed by this program.

USAGE

Program PGM5 is invoked during operation of the following Root Executive Sequences (NCSS Filetype = EXEC):

EXEC Sequence	Data Base
E22E	Instructor Master

INPUT FORMATS

Input record format for PGM5 is found on the following diagram titled "Instructor Data Base Weekly Time Card Format".

The User is terminal prompted for the time card processing date. This date takes the format YYMMDD, e.g., 750915. After verification of the User response, the date input is moved to the output record in positions 18 through 23, inclusive. This date will appear in every output transaction in the same positions.

INSTRUCTOR DATA BASE WEEKLY TIME CARD FORMAT

IBM Federal Systems Division, Space Systems Center, Huntsville, Alabama

INSTRUCTOR FILE

80-Column Card Punch Layout

NAME: INSTRUCTOR TIME CARD

WORK PACKAGE

WRITTEN AS

PUNCH AS

CARD PUNCH POINT: YES / NO

VERIFY: YES / NO

DATE DUE OUT

SHEET OF

INSTRUCTOR TIME(S), AS REQUIRED = $XX_{A}X$

A = LOCATION OF ASSUMED DECIMAL POINT

ID	SCH	CDP	A1	A2	A3	A4	A5	A6	A7	B1	B2	B3	B4	B5	B6	B6A

955-1115-0

OUTPUT FORMAT FOR PGM5

Logical Record Length: 121 Bytes

PL/1 Base Data Structure Name: RTRANS

Output PL/1 Filename: TRANS

Start Record Position	Byte Length	PL/1 Identifier	Format	Description
1	17	R17	A17	Direct copy of the first 17 chars of the input time card
18	6	RD	16	Time card transaction date in format YYMMDD as provided by the User terminal

24/121 = Repeating 14 groups of the following seven (7) character sequence:

1-3	TTT	A3	Instructor time type; A1 through A7, B1 through B6, and B6A are legal.
4-7	RTIM	F4.1	Time expended in the indicated time type

NOTE - As many time-type groups as are required to contain the input time card data will be used in any single output transaction. The remainder, if any, will be left blank.

```
PL/I MODULE        (LIST,NAMSQ,SOURCSEQ)

 CARD NO      ****                          CONTENTS                                  ****

   1          OPTION PACKIF,COMPRESS
   2          PGM5:   /* INSTRUCTOR FILE TIME CARD PROCESSOR: FRI/25/APR/75 */
   3                  PROC OPTIONS (MAIN) ;
   4
   5          DECLARE NC          FIXED BIN STATIC INIT ( 0 ),
   6                  NT          FIXED BIN STATIC INIT ( 0 ),
   7                  NER         FIXED BIN STATIC INIT ( 0 ),
   8                  NTER        FIXED BIN STATIC INIT ( 0 ),
   9                  CARD        CHAR (80) STATIC,
  10                  F17         CHAR (17) DEF CARD POS (1),
  11                  F59         CHAR (59) DEF CARD POS (1),
  12                   REP         CHAR (80) STATIC,
  13                   M1          CHAR (65) STATIC INIT (
  14        '** CONVERSION ERROR ON INPUT CARD - XX,XXX - FIELD CONTAINS XXX' ),
  15                   FM1          PICTURE 'ZZ,ZZ9' DEF M1 POS (37),
  16                   FM2          CHAR (3) DEF M1 POS (61),
  17                  TIMS (14) CHAR (3) DEF CARD POS (18),
  18                  LDTE        FIXED DEC (8) STATIC,
  19                  ( J, K )    FIXED BIN STATIC,
  20                  CTIME       FIXED DEC (3,1) STATIC,
  21                  TYPE (14) CHAR (3) STATIC INIT
  22                       ( 'A1 ', 'A2 ', 'A3 ', 'A4 ', 'A5 ', 'A6 ', 'A7 ',
  23                         'B1 ', 'B2 ', 'B3 ', 'B4 ', 'B5 ', 'B6 ', 'B6A' ),
  24                  FILL        PICTURE 'Z,ZZ9' STATIC ;
  25
  26          DECLARE 1 RTRANS STATIC,
  27                    2 R17   CHAR (17),
  28                    2 RD    PICTURE '999999',
  29                    2 TFIELDS (14),
  30                      3 TTT       CHAR (3),
  31                      3 RTIM      PICTURE '99V.9',
  32                  TCLR        CHAR (98) BASED (PC) ;
  33
  34          STARTERRUPP:
  35                  OPEN FILE (CARDS) RECORD INPUT,
  36                       FILE (ETRANS) RECORD OUTPUT,
  37                       FILE (TRANS) RECORD OUTPUT ;
  38                  ON ENDFILE (CARDS) GO TO WRAPUP ;
  39                  PC = ADDR ( TFIELDS ) ;
  40
  41          GET_UPDATE_DATE:
  42                  ON CONVERSION BEGIN ;
  43                       DISPLAY ( '** CONVERSION ERROR @ ' || ONCODE ) ;
  44                            GO TO DCC1 ;
  45                       END ;
  46
  47          DCC1:   DISPLAY ( '** ENTER TRANSACTION DATE - YYMMDD - 6 DIGITS' )
  48                       REPLY ( REP ) ;
  49                  J = INDEX ( REP, '  ' ) - 1 ;
  50                 IF ( J <= 0 ) THEN GO TO BUST_DATE1 ;
  51                  GET STRING ( SUBSTR(REP,1,J) ) EDIT
  52                       (LDTE) ( F(J) ) ;
  53                  IF ( LDTE < 750000 ) THEN GO TO BUST_DATE2 ;
  54                       RD = LDTE ;
  55                            GO TO MAINLINE ;
  56
  57          BUST_DATE1:
  58                  DISPLAY ( '** DATE FORMAT INDETERMINATE, TRY AGAIN' ) ;
  59                       GO TO DCC1 ;
  60          BUST_DATE2:
  61                  DISPLAY ( '** DATE VALUE LESS THAN 75XXXX, TRY AGAIN' ) ;
  62                       GO TO DCC1 ;
  63
  64          MAINLINE:
  65                  ON CONVERSION BEGIN ;
  66                       NER = NER + 1 ;
  67                       FM1 = NC ;
  68                       FM2 = ONSOURCE ;
  69                       DISPLAY ( M1 ) ;
  70                       DISPLAY ( '** FIRST 59 CHARACTERS OF CARD IMAGE ARE =' ) ;
  71                       DISPLAY ( '**        --' || F59 || '--' ) ;
  72                       WRITE FILE (ETRANS) FROM (CARD) ;
  73                       NTER = NTER + 1 ;
  74                       DISPLAY( '** TRANSACTION WILL BE IGNORED' ) ;
  75                            GO TO R1 ;
  76                       END ;
  77
  78          R1:     READ FILE (CARDS) INTO (CARD) ;
  79                  NC = NC + 1 ;
  80                  TCLR = ' ' ;
  81                  R17 = F17 ;
  82                  J = 0 ;
  83
  84          TLOOP:       DO K = 1 TO 14 ;
  85                       IF ( TIMS(K) = '   ' ) THEN GO TO ETLOOP ;
  86                       GET STRING ( TIMS(K) ) EDIT
  87                            (CTIME) ( F(3,1) ) ;
  88                       J = J + 1 ;
  89                       TTT (J) = TYPE (K) ;
  90                       RTIM (J) = CTIME ;
  91          ETLOOP:      END TLOOP ;
  92
  93                       WRITE FILE (TRANS) FROM (RTRANS) ;
  94                       NT = NT + 1 ;
  95                            GO TO R1 ;
  96
  97          WRAPUP:
  98                  DISPLAY (
  99        '** INSTRUCTOR TIME TRANSACTION PROCESSING HAS COMPLETED' ) ;
 100                  FILL = NC ;
 101                  DISPLAY ( '** TOTAL INPUT CARDS READ = ' || FILL ) ;
 102                  FILL = NT ;
 103                  DISPLAY ( '** TOTAL VALID OUTPUT TRANSACTIONS PROCESSED = '
 104                       || FILL ) ;
 105                  FILL = NER ;
 106                  DISPLAY ( '** TOTAL NUMBER CARD ERRORS = ' || FILL ) ;
 107                  FILL = NTER ;
 108                  DISPLAY (
 109        '** TOTAL NUMBER CARD IMAGES IN ERROR-TRANSACTION FILE = ' || FILL ) ;
```

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PGM5- FLOWCHARTS

CARD NO	****	CONTENTS	****
110			00010900
111	WINDUP:		00011000
112		CLOSE FILE (CARDS),	00011100
113		FILE (ETRANS),	00011200
114		FILE (TRANS) ;	00011300
115			00011400
116		END PGM5 ;	00011500

CHART TITLE - PROC PGM5 OPTIONS (MAIN)

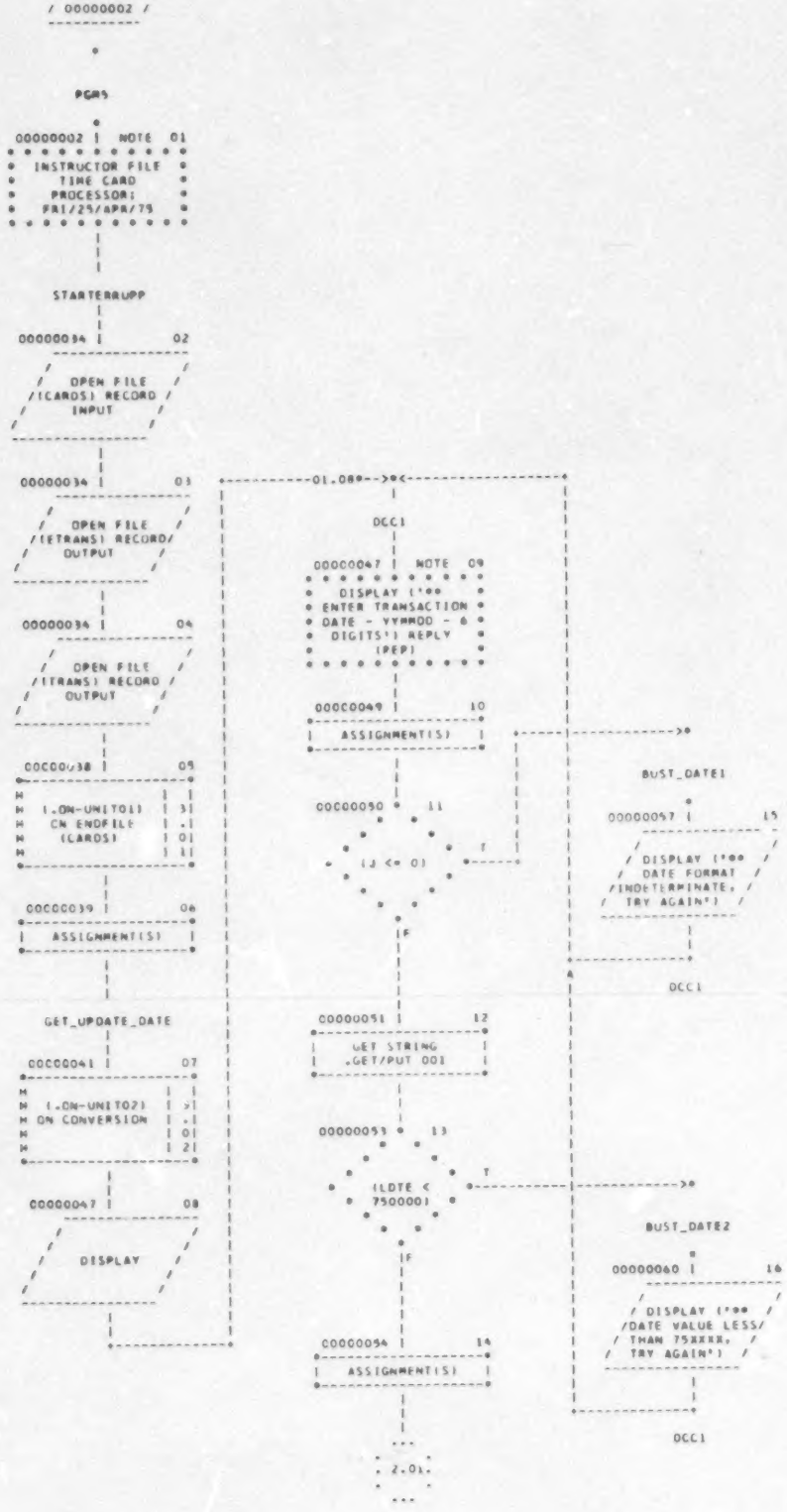


CHART TITLE - PROC PGM5 OPTIONS (MAIN)

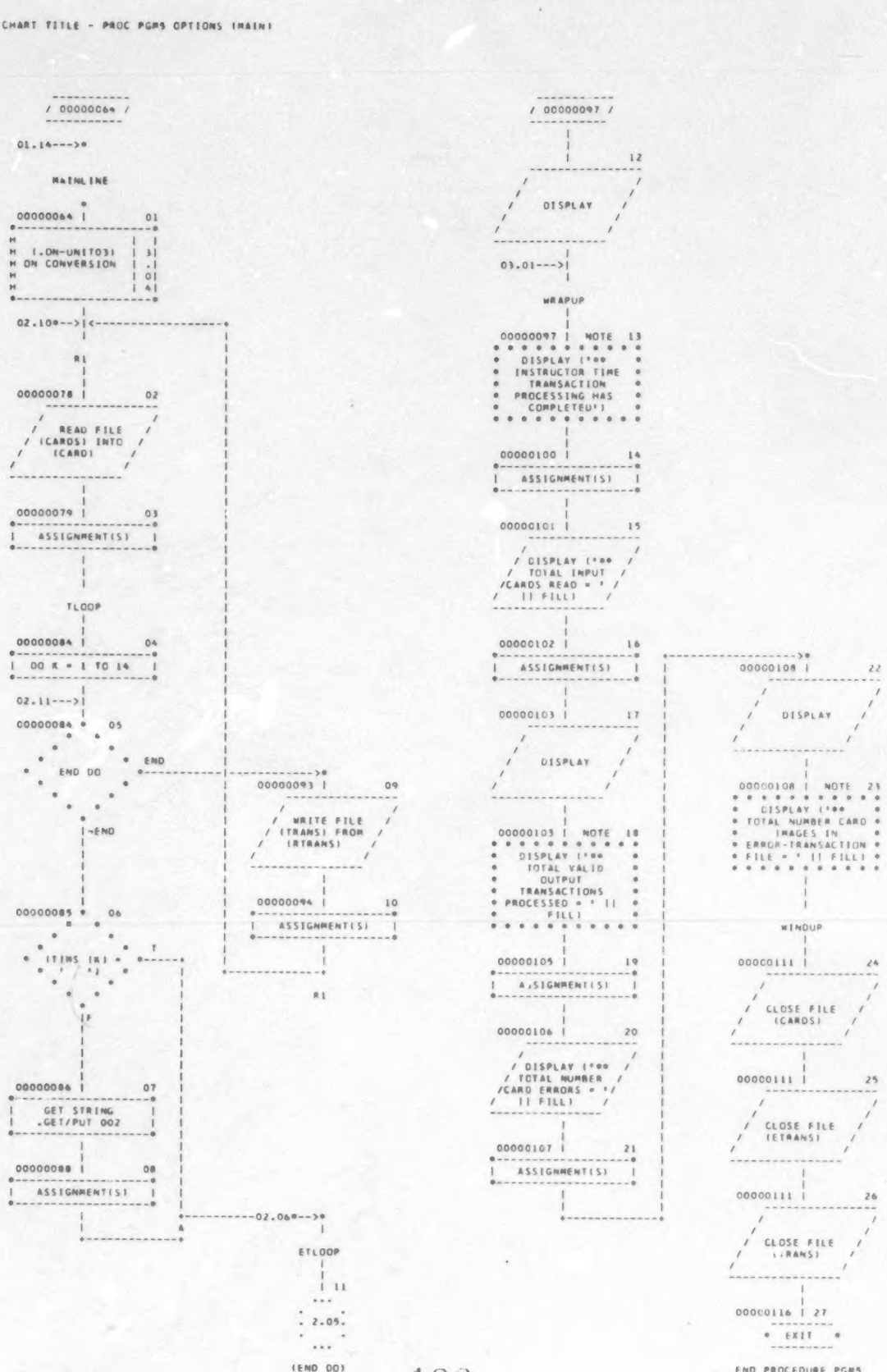


CHART TITLE - ON-UNIT ACTION BLOCKS

```
      ----------
     / 00000038 /
     ----------

01.05--->*

.ON-UNIT01

          *
00000038 |  NOTE  01
* * * * * * * * * * *
*   ON ENDFILE      *
*     (CARDS)       *
* * * * * * * * * * *
          |
          |
          |
         ...
       .     .
       . 2.13.
       .     .
         ...

      WRAPUP

   END .ON-UNIT01


      ----------
     / 00000042 /
     ----------

01.07--->*

.ON-UNIT02

          *
00000042 |  NOTE  02
* * * * * * * * * * *
*   ON CONVERSION   *
*                   *
*       BEGIN       *
* * * * * * * * * * *
          |
          |
00000043 |         03
     ----------------
    /                /
   / DISPLAY ('**  /
  /  CONVERSION   /
 / ERROR # ' ||  /
/    ONCODE)    /
----------------
          |
          |
          |
         ...
       .     .
       . 1.09.
       .     .
         ...

       DCC1

   END .ON-UNIT02
```

```
      ----------
     / 00000065 /
     ----------

02.01--->*

.ON-UNIT03

          *
00000065 |  NOTE  04
* * * * * * * * * * *
*   ON CONVERSION   *
*                   *
*       BEGIN       *
* * * * * * * * * * *
          |
          |
00000066 |         05
*-----------------*
|   ASSIGNMENT(S)  |
*-----------------*
          |
          |
00000069 |         06
     ----------------
    /               /
   /               /
  / DISPLAY (M1)  /
 /               /
/               /
----------------
          |
          |
00000070 |         07
     ----------------
    / DISPLAY ('**  /
   /   FIRST 59    /
  / CHARACTERS OF /
 /CARD IMAGE ARE /
/      *')      /
----------------
          |
          |
00000071 |         08
     ----------------
    /               /
   / DISPLAY ('**  /
  / --' || F59 || /
 /     '--')     /
/               /
----------------
          |
          |
00000072 |         09
     ----------------
    /               /
   /  WRITE FILE   /
  / (FTRANS) FROM /
 /    (CARD)     /
/               /
----------------
          |
          |
00000073 |         10
*-----------------*
|   ASSIGNMENT(S)  |
*-----------------*
          |
          |
00000074 |         11
     ----------------
    /               /
   / DISPLAY ('**  /
  /  TRANSACTION  /
 /    WILL BE    /
/   IGNORED')   /
----------------
          |
          |
          |
         ...
       .     .
       . 2.02.
       .     .
         ...

       R1

   END .ON-UNIT03
```

CHART TITLE - GET/PUT, FORMAT STATEMENTS

```
00000051          .GET/PUT 001  GET STRING (SUBSTR (REP,I,J)) EDIT (!DTE) (F (J))
00000086          .GET/PUT 002  GET STRING (TIMS (K)) EDIT (CTIME) (F (3,1))
```

PROGRAM NAME: PGM7

TYPE: PL/1

SYSTEM: Data Base Maintenance

NCSS FILENAME/FILETYPE: PGM7/PLI

A listing of the program source code and machine run flowcharts are included in this document on the pages following.

FUNCTION

Use of the RAMIS data maintenance function DUMP produces an 80-Byte (card image) output file, regardless of the length of the individual RAMIS records. As many card images per RAMIS record as required will be used to represent RAMIS records longer than 80 Bytes.

For the Instructor data base, the field positions in the output of the DUMP are indicated in the following sample RAMIS output:

± EBCDIC RECORD LENGTH 8Ø CHARACTERS

FIELD NAME	FORMAT	LENGTH
ID	A 9	9
NME	A 16	16
IN	A 2	2
RATE	A 5	5
SCH	A 4	4
REP	I 4	4
ROT	I 4	4
PNEC	A 4	4
SNEC	A 4	4
CDP	A 4	4
TTL	A 16	16
ONB	A 1	1
LDTE	I 6	6
		X 1
TT	A 3	3
CW	F4 .1	4
QTD	F5 .1	5
YTD	F6 .1	6

Note that one slack byte (X) was included on the first card to prevent split of a field across logical record boundaries. Also, only 18 Bytes are required on the 2nd card image.

Inputs to RAMIS data maintenance functions must be singular records. The function of program PGM7 is to convert the multiple dump format to a single logical record per pair of input records.

For the Instructor file, using input as indicated above, the output record is composed as follows:

Logical Record Length:	97 Bytes
PL/1 Base Data Structure Name:	ROUT
Output PL/1 Filename:	TRANS
First 79 Characters = Direct copy from input card image number 1	
Characters 80 through 97, Inclusive:	Direct copy from first 18 characters of input card image number 2

USAGE

Program PGM7 is invoked during operation of the following Root Executive Sequences (NCSS Filetype = EXEC):

EXEC Sequence	Data Base
E32	Instructor
E27	Instructor

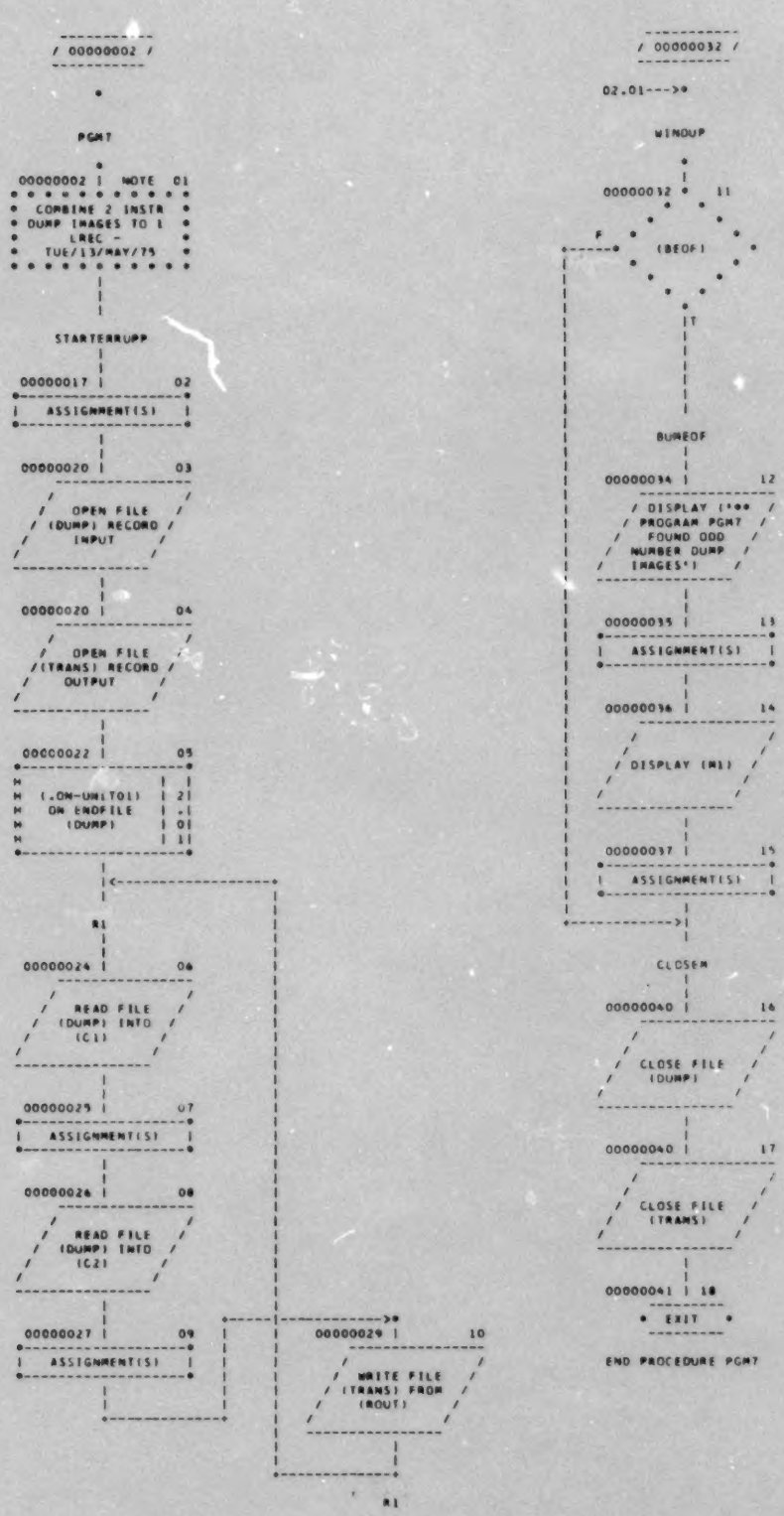
INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PGM7- FLOWCHARTS

PL/I MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
OPTION PACKIF,COMPRESS
 PGM7:   /* COMBINE 2 INSTR DUMP IMAGES TO 1 LREC - TUE/13/MAY/75 */
         PROC OPTIONS (MAIN) ;

 DECLARE BUFF (259) CHAR (1) STATIC,
         C1        CHAR (80) BASED (P1),
         C2        CHAR (80) BASED (P2),
         ROUT      CHAR (97) BASED (P3),
         BEOF      BIT (1) STATIC INIT ( '0'B ),
         NC        FIXED BIN STATIC INIT ( 0 ),
         NN        FIXED BIN (31,0) STATIC INIT ( 0 ),
         M1        CHAR (39) STATIC INIT (
 '**      TOTAL DUMP IMAGE COUNT = XX,XXX' ),
         FM1       PICTURE 'ZZ,ZZ9' DEF M1 POS (34),
         LBL       CHAR (5) STATIC INIT ( '*    ' ) ;

 STARTERRUPP:
         P1, P3 = ADDR ( BUFF(1) ) ;
         P2 = ADDR ( BUFF(80) ) ;
         OPEN FILE (DUMP) RECORD INPUT,
              FILE (TRANS) RECORD OUTPUT ;
         ON ENDFILE (DUMP) GO TO WINDUP ;

 R1:     READ FILE (DUMP) INTO (C1) ;
              BEOF = '1'B ;
         READ FILE (DUMP) INTO (C2) ;
         BEOF = '0'B ;
              NC = NC + 2 ;
              WRITE FILE (TRANS) FROM (ROUT) ;
              GO TO P1 ;

 WINDUP: IF ( BEOF ) THEN
 BUMEOF:      DO ;
              DISPLAY ('** PROGRAM PGM7 FOUND ODD NUMBER DUMP IMAGES');
              FM2 = NC ;
              DISPLAY ( M1 ) ;
                   NN = 4 ;
              END BUMEOF ;

 CLOSEM: CLOSE FILE (DUMP), FILE (TRANS) ;
              END PGM7 ;
```

CHART TITLE - PROC PGM7 OPTIONS (MAIN)



AUTOFLOW CHART SET - FINAL PL/I -PGM7- FLOWCHART. PAGE 2

CHART TITLE - ON-UNIT ACTION BLOCKS

```
      -----------
     / 00000022 /
     -----------

 01.09--->*

.ON-UNIT01

          *
 00000022 |  NOTE  01
* * * * * * * * * * *
* ON ENDFILE (DUMP) *
* * * * * * * * * * *
          |
          |
          |
         ...
        .     .
        . 1.11.
        .     .
         ...

       WINDUP

   END .ON-UNIT01
```

PROGRAM NAME: PGM8

TYPE: PL/1

SYSTEM: Data Base Maintenance

NCSS FILENAME/FILETYPE: PGM8/PLI

A listing of the program source code and machine run flowcharts are included in this document on the pages following.

FUNCTION

Following a standard RAMIS data maintenance DUMP of the Instructor data base, PGM8 is used to cause a display of each of the records of the data base.

USAGE

Program PGM8 is invoked during operation of the following Root Executive Sequences (NCSS Filetype = EXEC):

EXEC Sequence	Data Base
E30	Instructor Master
E35	Instructor Scratch

INPUT FORMAT

For the Instructor data base, the field positions in the output of the RAMIS DUMP are indicated in the following sample RAMIS output:

± EBCDIC RECORD LENGTH 8Ø CHARACTERS

FIELD NAME	FORMAT		LENGTH
ID	A	9	9
NME	A	16	16
IN	A	2	2
RATE	A	5	5
SCH	A	4	4
REP	I	4	4
ROT	I	4	4
PNEC	A	4	4
SNEC	A	4	4
CDP	A	4	4
TTL	A	16	16
ONB	A	1	1
LDTE	I	6	6
			X 1
TT	A	3	3
CW	F4	.1	4
QTD	F5	.1	5
YTD	F6	.1	6

198

Note that one slack byte has been included on the first card image at position (X); this prevents split of a field across distinct logical record boundaries. Also, only 18 Bytes of the second card image are required to complete the RAMIS record.

OUTPUT RECORDS

Output of PGM8 is to the User terminal. Two (2) input card images are used in the construction of an output image. This is the minimum count. For each Course assigned to each Instructor, there will be an input record pair.

Output is composed to fit on the 80-character display face; essentially, an entire screen is required to display the data associated with a single Instructor.

A sample of the output is indicated in the following illustration titled "Sample Terminal Output of Program PGM8".

SAMPLE TERMINAL RUN OF PROGRAM PGM8

```
    SCHOOL: COMM                    RATE: RM2          REPORT: 1272     PNEC:
INSTRUCTOR: RA SNYDER                 ID: 175424494    ROTATE:  676     SNEC: 95Ø6
-------------------------------------------------------------------------------------
                          !                        !                        !                        !
    COURSE CDP: 538R      ! COURSE CDP: 512F       ! COURSE CDP: 512H       ! COURSE CDP: 512K       !
    ONBOARD: YES          ! ONBOARD: YES           ! ONBOARD: YES           ! ONBOARD: YES           !
    LUDATE:   / Ø/        ! LUDATE:   / Ø/         ! LUDATE:   / Ø/         ! LUDATE:   / Ø/         !
    -CUR--QTD---YTD--!    -CUR--QTD---YTD--!    -CUR--QTD---YTD--!    -CUR--QTD---YTD--!
A1    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
A2    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
A3    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
A4    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
A5    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
A6    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
A7    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
                      !                        !                        !                        !
B1    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
B2    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
B3    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
B4    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
B5    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
B6    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
B6A   .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!    .Ø    .Ø    .Ø!
```

INPUT LISTING   AUTOFLOW CHART SET - FINAL   PL/I -PGM8- FLOWCHARTS

PL/I MODULE   (LIST,NAMSQ,SOURCSEQ)

CARD NO   ****   CONTENTS   ****

```
OPTION PACKIF,COMPRESS
PGM8:   /* PRINT INSTR. FILE FROM DISK DUMP - TUE/6/MAY/75 */
        PROC OPTIONS (MAIN) ;

DECLARE 1 CARD1    STATIC,
          2 ID     CHAR (9),
          2 NME    CHAR (16),
          2 IN     CHAR (2),
          2 RATE   CHAR (5),
          2 SCH    CHAR (4),
          2 REP    CHAR (4),
          2 ROT    CHAR (4),
          2 PNEC   CHAR (4),
          2 SNEC   CHAR (4),
          2 CDP    CHAR (4),
          2 TTL    CHAR (16),
          2 ONR    CHAR (1),
          2 LDTE   CHAR (6),
          2 FIL1   CHAR (1),
        1 CARD2    STATIC,
          2 TT     CHAR (3),
          2 CT     CHAR (4),
          2 QTD    CHAR (5),
          2 YTD    CHAR (6),
          2 FIL2   CHAR (62) ;

DECLARE M1          CHAR (63) STATIC INIT (
'** EOF DETECTED BEFORE ALL TIMES READ - CARD COUNT IS XX,XXX **' ),
        FILLM1      PICTURE 'ZZ,ZZ9' DEF M1 POS (55),
        M3          CHAR (59) STATIC INIT (
'** SEQUENCING ERROR # X IN PROCESSING INPUT RECORD # XX,XXX' ),
        FILLM3A     CHAR (1) DEF M3 POS (23),
        FILLM3B     PICTURE 'ZZ,ZZ9' DEF M3 POS (54),
        M2          CHAR (56) STATIC INIT (
'** COURSE CDP = XXXX, COURSE TITLE = "XXXXXXXXXXXXXXX **' ),
        FILLM2A     CHAR (4) DEF M2 POS (17),
        FILLM2B     CHAR (16) DEF M2 POS (38) ;

DECLARE OLDID       CHAR (9) STATIC INIT ( ' ' ),
        L1          CHAR (78) STATIC INIT (
'    SCHOOL: XXXX                 RATE: XXXXX        REPORT: XX/XX   PN
EC: XXXX' ),
        LSCH        CHAR (4) DEF L1 POS (13),
        LRTE        CHAR (5) DEF L1 POS (41),
        LREP        CHAR (5) DEF L1 POS (61),
        LPNEC       CHAR (4) DEF L1 POS (75),
        L2          CHAR (78) STATIC INIT (
'INSTRUCTOR: XX XXXXXXXXXXXXXXXX      ID: XXXXXZXXX   ROTATE: XX/XX   SN
EC: XXXX' ),
        LNM         CHAR (16) DEF L2 POS (16),
        LIN         CHAR (2)  DEF L2 POS (13),
        LID         CHAR (9)  DEF L2 POS (41),
        LROT        CHAR (5)  DEF L2 POS (61),
        LSNEC       CHAR (4)  DEF L2 POS (75),
        L3          CHAR (78) STATIC INIT ( (78)'-' ),
        LBLANK      CHAR (5) STATIC INIT ( ' ' ),
        L8          CHAR (19) STATIC INIT
             ( ' -CUR--QTD---YTD--|' ) ;

DECLARE OUTSEG (20,4) CHAR (19) BASED (P1),
        OUTROW (20) CHAR (76) BASED (P2),
        OUTCLR      CHAR (1520) STATIC INIT ( ' ' ),
        BUILD       CHAR (19) STATIC INIT
             ( '                 |' ),
             BCT  CHAR (4) DEF BUILD POS (2),
             BQT  CHAR (5) DEF BUILD POS (7),
             BYT  CHAR (6) DEF BUILD POS (13),
        BL4         CHAR (19) STATIC INIT
             ('                  |' ),
             FBL4 CHAR (16) DEF BL4 POS (2),
        BL5         CHAR (19) STATIC INIT
             (' COURSE CDP: XXXX |' ),
             FBL5 CHAR (4) DEF BL5 POS (14),
        BL6         CHAR (19) STATIC INIT
             (' ONBOARD: XXX     |' ),
             FBL6 CHAR (3) DEF BL6 POS (11),
        BL7         CHAR (19) STATIC INIT
             (' LUDATE: MM/DD/YY |' ),
             L7MM CHAR (2) DEF BL7 POS (10),
             L7DD CHAR (2) DEF BL7 POS (13),
             L7YY CHAR (2) DEF BL7 POS (16),
        GETDATE     CHAR (6) STATIC,
             GMM  CHAR (2) DEF GETDATE POS (3),
             GDD  CHAR (2) DEF GETDATE POS (5),
             GYY  CHAR (2) DEF GETDATE POS (1) ;

DECLARE BUMEOF      BIT (1) STATIC INIT ( '0'B ),
        TTYPE (20) CHAR (3) STATIC INIT ( (5)(1)'   ',
             'A1 ', 'A2 ', 'A3 ', 'A4 ', 'A5 ', 'A6 ', 'A7 ', '   ',
             'B1 ', 'B2 ', 'B3 ', 'B4 ', 'B5 ', 'B6 ', 'B6A' ),
        N           FIXED BIN STATIC INIT ( 0 ),
        LL          STATIC,
        ( J, NC ) FIXED BIN STATIC,
        OLINE       CHAR (79) STATIC,
        ROUTE       LABEL ;

PUT18:  FORMAT ( (N)A(18) ) ;


STARTERRUPP:
        OPEN FILE (DUMP) RECORD INPUT ;
        ON ENDFILE (DUMP) GO TO WINDUP ;
        P1, P2 = ADDR (OUTCLR) ;

R1:     READ FILE (DUMP) INTO (CARD1) ;
        READ FILE (DUMP) INTO (CARD2) ;
        NC = NC + 2 ;
        IF ( ID = OLDID ) THEN GO TO MORE_THIS_INSTRUCTOR ;
        IF ( N > 0 ) THEN
```

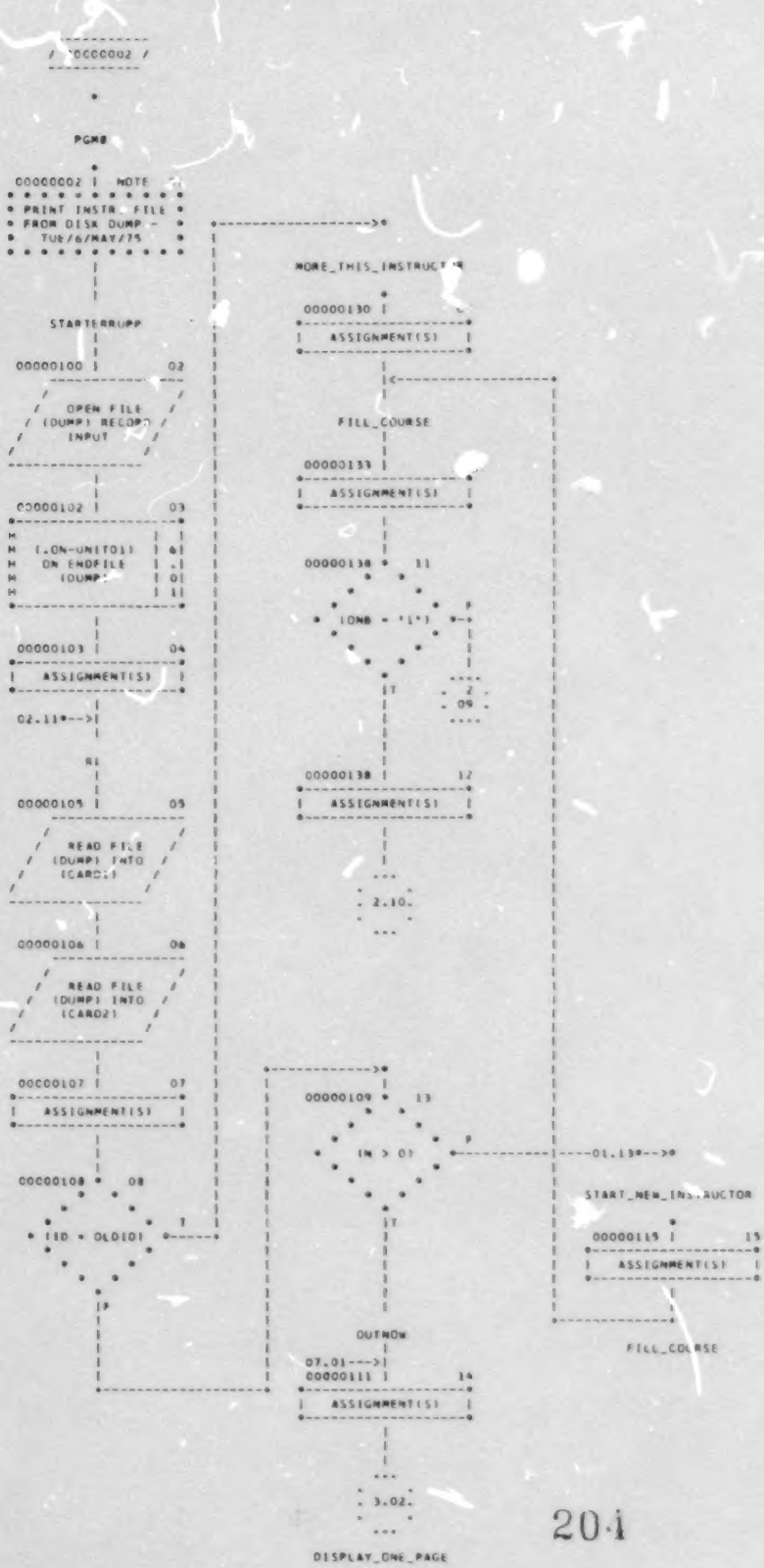
```
CARD NO      ****                    CONTENTS                    ****

  110        OUTNOW:      DO ;                                          00010900
  111                     ROUTE = START_NEW_INSTRUCTOR ;                00011000
  112                          GO TO DISPLAY_ONE_PAGE ;                 00011100
  113                     END OUTNOW ;                                  00011200
  114                                                                   00011300
  115        START_NEW_INSTRUCTOR:                                      00011400
  116                N = 1 ;                                            00011500
  117                OLDID = ID ;                                       00011600
  118                LSCH = SCH ;                                       00011700
  119                LRTE = RATE ;                                      00011800
  120                LREP = REP ;                                       00011900
  121                LPNEC = PNEC ;                                     00012000
  122                LNM = NME ;                                        00012100
  123                LIN = IN ;                                         00012200
  124                LID = ID ;                                         00012300
  125                LROT = ROT ;                                       00012400
  126                LSNEC = SNEC ;                                     00012500
  127                OUTCLR = ' ' ;                                     00012600
  128                     GO TO FILL_COURSE ;                           00012700
  129                                                                   00012800
  130        MORE_THIS_INSTRUCTOR:                                      00012900
  131                N = N + 1 ;                                        00013000
  132                                                                   00013100
  133        FILL_COURSE:                                               00013200
  134                FBL4 = TTL ;                                       00013300
  135                     OUTSEG (1,N) = BL4 ;                          00013400
  136                FBL5 = CDP ;                                       00013500
  137                     OUTSEG (2,N) = BL5 ;                          00013600
  138                IF ( ONB = '1' ) THEN FBL6 = 'YES' ;               00013700
  139                                  ELSE FBL6 = 'NO ' ;              00013800
  140                     OUTSEG (3,N) = BL6 ;                          00013900
  141                GETDATE = LDTE ;                                   00014000
  142                          L7MM = GMM ;                             00014100
  143                          L7DD = GDD ;                             00014200
  144                          L7YY = GYY ;                             00014300
  145                     OUTSEG (4,N) = BL7 ;                          00014400
  146                OUTSEG (5,N) = L8 ;                                00014500
  147                                                                   00014600
  148                IF ( TT = '   ' ) THEN GO TO R1 ;                  00014700
  149                                                                   00014800
  150                BUMEOF = '1'B ;                                    00014900
  151                LL = 5 ;                                           00015000
  152        GET_TIMES:   DO J = 1 TO 15 ;                             00015100
  153                     IF ( ID ¬= OLDID ) THEN GO TO ERROR1 ;        00015200
  154                     LL = LL + 1 ;                                 00015300
  155                                                                   00015400
  156                     IF ( LL = 13 ) THEN                           00015500
  157        FILL13:          DO ;                                     00015600
  158                          OUTSEG (13,N) = BL4 ;                    00015700
  159                               GO TO EGTIMS ;                      00015800
  160                          END FILL13 ;                             00015900
  161                                                                   00016000
  162                          BCT = CT ;                               00016100
  163                          BQT = QTD ;                              00016200
  164                          BYT = YTD ;                              00016300
  165                          OUTSEG (LL,N) = BUILD ;                  00016400
  166                                                                   00016500
  167                     IF ( LL >= 20 ) THEN GO TO CYCLE ;            00016600
  168                     READ FILE (DUMP) INTO (CARD1) ;               00016700
  169                     READ FILE (DUMP) INTO (CARD2) ;               00016800
  170                     NC = NC + 2 ;                                 00016900
  171        EGTIMS:      END GET_TIMES ;                               00017000
  172                                                                   00017100
  173        CYCLE:   BUMEOF = '0'B ;                                  00017200
  174                IF ( N < 4 ) THEN GO TO R1 ;                       00017300
  175                                                                   00017400
  176        JUST_FILLED_A_PAGE:                                        00017500
  177                ROUTE = RECYCLE ;                                  00017600
  178                     GO TO DISPLAY_ONE_PAGE ;                      00017700
  179                                                                   00017800
  180        RECYCLE:                                                   00017900
  181                N = 0 ;                                            00018000
  182                     GO TO R1 ;                                    00018100
  183                                                                   00018200
  184                                                                   00018300
  185        DISPLAY_ONE_PAGE:                                          00018400
  186                DISPLAY ( LBLANK ) ;      /* PAGE-CLEAR GOES HERE,     */ 00018500
  187                DISPLAY ( LBLANK ) ;      /*     WHENEVER AVAILABLE... */ 00018600
  188                DISPLAY ( L1 ) ;                                   00018700
  189                DISPLAY ( L2 ) ;                                   00018800
  190                DISPLAY ( L3 ) ;                                   00018900
  191                                                                   00019000
  192        TUBE:        DO J = 1 TO 20 ;                              00019100
  193                     DISPLAY ( TTYPE(J) || OUTROW(J) ) ;           00019200
  194                     END TUBE ;                                    00019300
  195                                                                   00019400
  196                          GO TO ROUTE ;  /* VARIABLE EXIT ROUTER */ 00019500
  197                                                                   00019600
  198                                                                   00019700
  199        ERROR1: FILLM3A = '1' ;                                    00019800
  200                     GO TO BUSTED ;                                00019900
  201        ERROR2: FILLM3A = '2' ;                                    00020000
  202                                                                   00020100
  203        BUSTED: FILLM3B = NC ;                                     00020200
  204                     DISPLAY ( LBLANK ) ;                          00020300
  205                     DISPLAY ( M3 ) ;                              00020400
  206                     DISPLAY ( '**        FOR FOLLOWING INSTRUCTOR...' ) ; 00020500
  207                                                                   00020600
  208                     DISPLAY ( LBLANK ) ;                          00020700
  209                     DISPLAY ( L1 ) ;                              00020800
  210                     DISPLAY ( L2 ) ;                              00020900
  211                     DISPLAY ( L3 ) ;                              00021000
  212                                                                   00021100
  213                FILLM2A = CDP ;                                    00021200
  214                FILLM2B = TTL ;                                    00021300
  215                     DISPLAY ( LBLANK ) ;                          00021400
  216                     DISPLAY ( M2 ) ;                              00021500
  217                                                                   00021600
  218                DISPLAY ( LBLANK ) ;                               00021700
  219                DISPLAY ( '** ABNORMAL DISPLAY-PROGRAM TERMINATION FOLLOWS' ) ; 00021800
  220                DISPLAY ( LBLANK ) ;                               00021900
  221                                                                   00022000
```

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PGM8- FLOWCHARTS

CARD NO **** CONTENTS ****

```
                GO TO CLOSEM ;
                                                                  00022100
                                                                  00022200
                                                                  00022300
        WINDUP:                                                   00022400
                IF ( BUMEOF ) THEN                                00022500
        BEOF:       DO ;                                          00022600
                    FILLM1 = NC ;                                 00022700
                        DISPLAY ( M1 ) ;                          00022800
                    FILLM2A = CDP ;                               00022900
                    FILLM2B = TTL ;                               00023000
                        DISPLAY ( M2 ) ;                          00023100
                    DISPLAY ( LBLANK ) ;                          00023200
                    DISPLAY ( L1 ) ;                              00023300
                    DISPLAY ( L2 ) ;                              00023400
                        GO TO CLOSEM ;                            00023500
                    END BEOF ;                                    00023600
                                                                  00023700
                IF ( N > 0 ) THEN                                 00023800
        ANOTHER:    DO ;                                          00023900
                    ROUTE = CLOSEM ;                              00024000
                        GO TO DISPLAY_ONE_PAGE ;                  00024100
                    END ANOTHER ;                                 00024200
                                                                  00024300
        CLOSEM:                                                   00024400
                CLOSE FILE (DUMP) ;                               00024500
                    END PGM8 ;                                    00024600
```

CHART TITLE - PROC PGMB OPTIONS (MAIN)



204

CHART TITLE - PROC PGMB OPTIONS (MAIN)

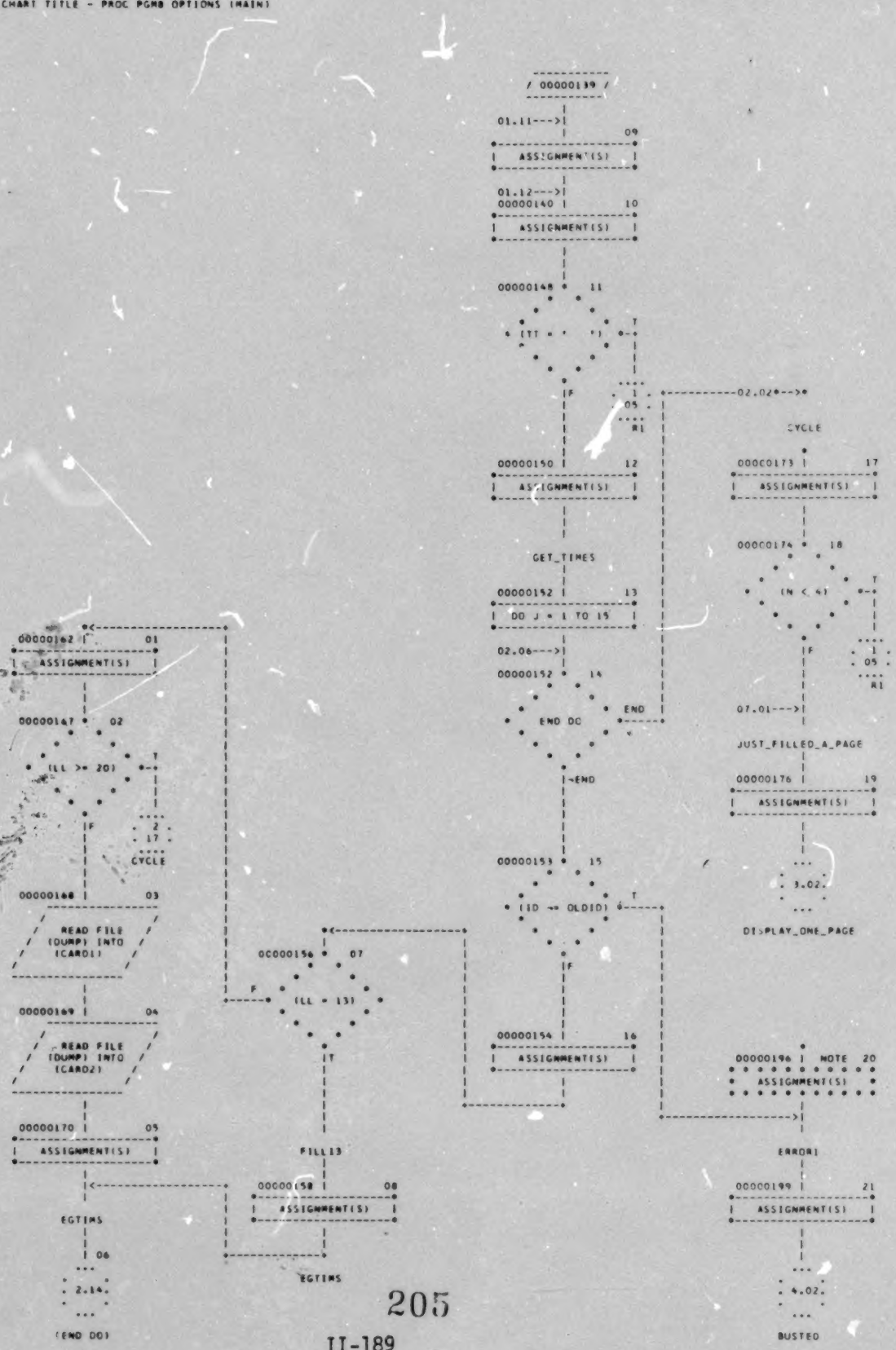


CHART TITLE - PROC PGM8 OPTIONS (MAIN)

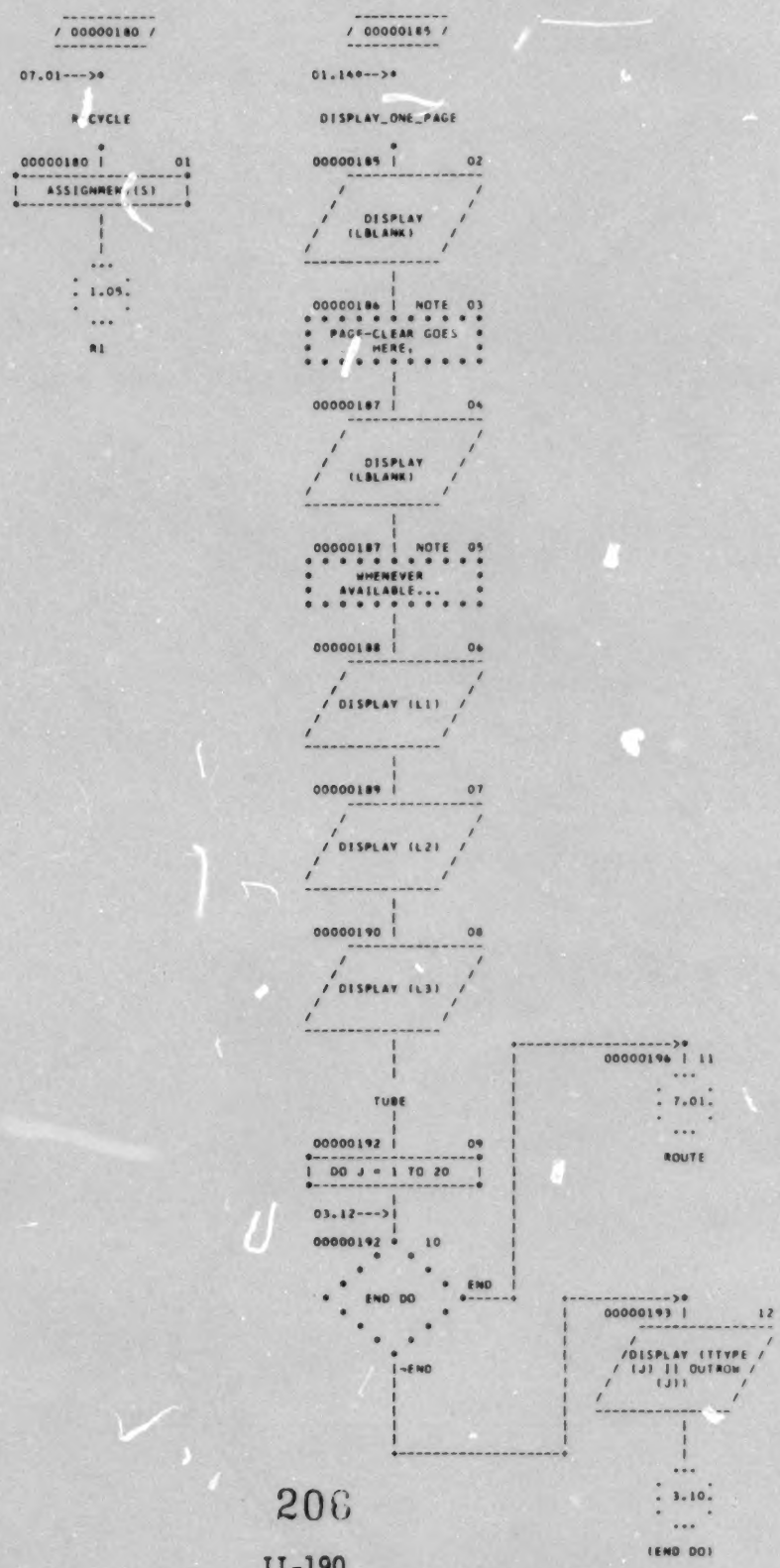


CHART TITLE - PROC PGM8 OPTIONS (MAIN)

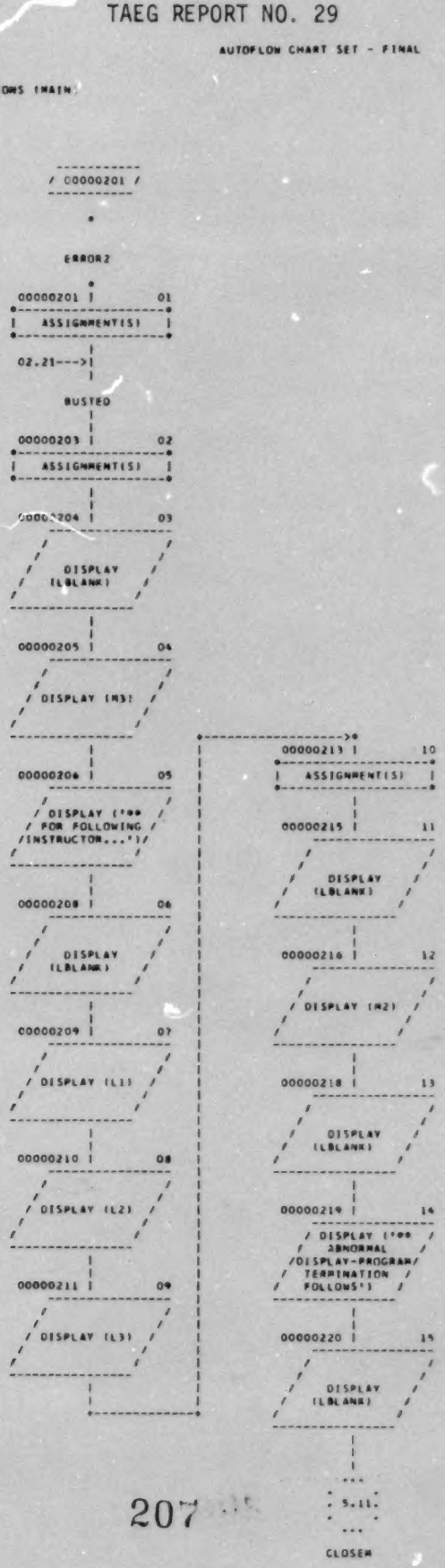


CHART TITLE - PROC PGM8 OPTIONS (MAIN)

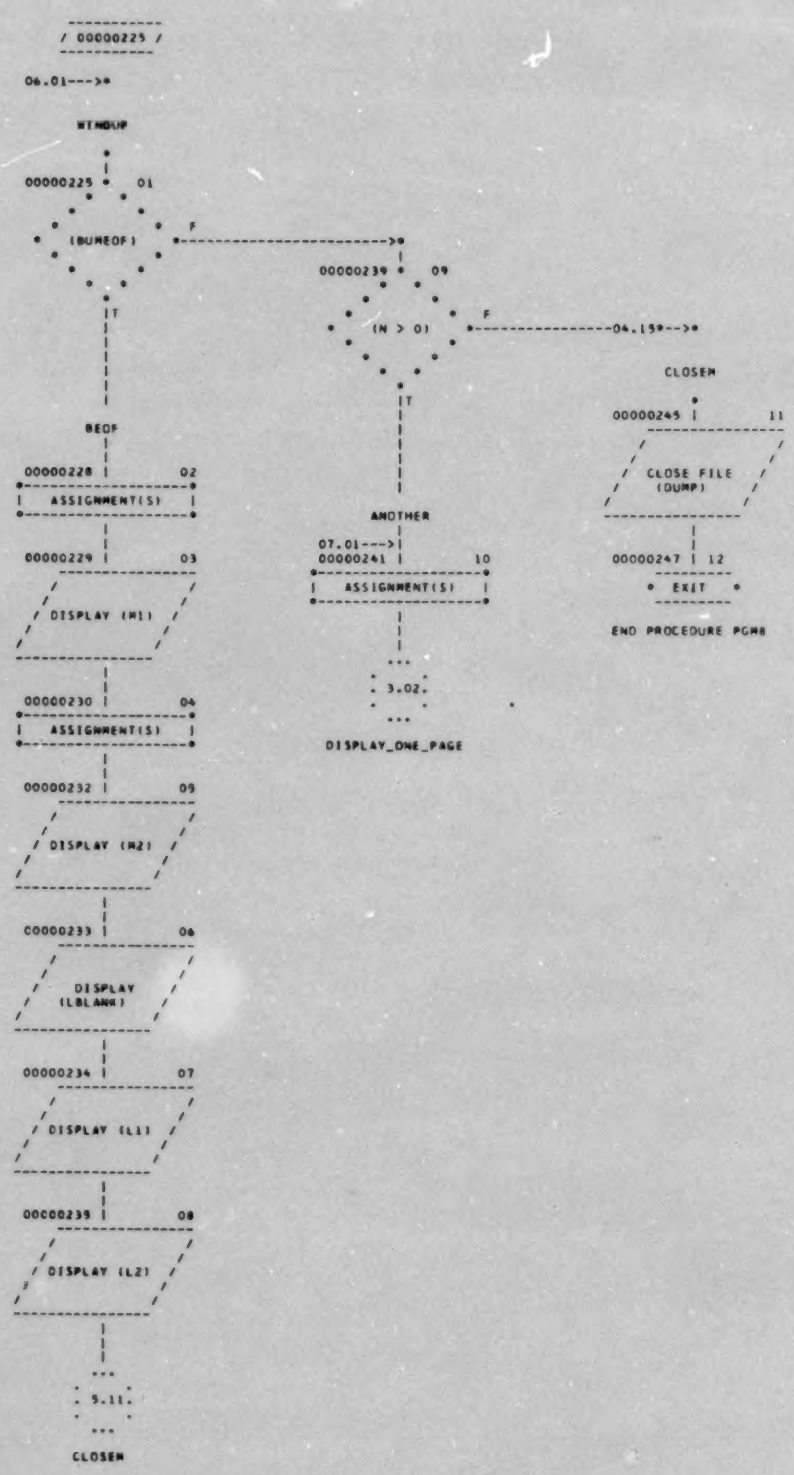


CHART TITLE - ON-UNIT ACTION BLOCKS

```
      ------------
      / 00000102 /
      ------------

 01.03--->*

.ON UNIT01

            *
 00000102 |  NOTE  01
* * * * * * * * * * *
* ON ENDFILE (DUMP) *
* * * * * * * * * * *
            |
            |
            |
          ...
        .     .
        . 5.01.
        .     .
          ...

        WINDUP

    END .ON-UNIT01
```

209

CHART TITLE - LABEL ASSIGNMENT CROSS REFERENCE

```
      ----------
     /  ROUTE   /
     ----------
          |
 03.11--->|
          |          01
*-------------------*
|      SCALAR       |
|  LABEL VARIABLE   |
*-------------------*
| SET AT  TO GO TO  |
|    1.14    1.15   |
| 00000111 START_NE |
|    2.19    3.01   |
| 00000177 RECYCLE  |
|    5.10    5.11   |
| 00000241 CLOSEM   |
*-------------------*
```

CHART TITLE - GET/PUT, FORMAT STATEMENTS

```
00000097 PUT18      FORMAT ((N) A (181))
```

PROGRAM NAME: PGM9B

TYPE: PL/1

SYSTEM: Data Base Maintenance

NCSS FILENAME/FILETYPE: PGM9B/PLI

A listing of the program source code and machine run flowcharts are included in this document on the pages following.

FUNCTION

PGM9B converts the data found on the Instructor data base Type -2- input cards to the format described below. Output records (transactions) from this program are subsequently processed into the data base by the RAMIS data maintenance REVISE function RE24/RAM (filename/filetype).

USAGE

Program PGM9B is invoked during operation of the following Root Executive Sequences (NCSS Filetype = EXEC):

EXEC Sequence	Data Base
E24	Instructor Master

INPUT FORMAT

Input to PGM9B is via cards; the input card format is found on the illustration following bearing the title "Instructor (Course) Data Base Type -2- Input Card Format".

Note that this input card type is used in loading both the Instructor and the Course data bases.

Input card column 53 contains a flag (ASN) which indicates whether the Instructor is currently assigned to the Course. Note that an Instructor can be qualified to teach a Course, but not assigned to teach the same Course. A character '1' in the ASN field indicates that the Instructor is currently assigned to teach; PGM9B will select only the current assignments for inclusion in the Instructor data base.

INSTRUCTOR (COURSE) DATA BASE TYPE -2- INPUT CARD FORMAT

INSTRUCTOR FILE

IBM Federal Systems Division, Space Systems Center, Huntsville, Alabama

NAME: TYPE -2- CARD

JOB NO

WORK PACKAGE

WRITTEN AS

PUNCH AS

CARD PUNCH PRINT: YES NO

VERIFY: YES NO

DATE DUE OUT

SHEET OF

80 Column Card Punch Layout

ID	NME	IN	SCH	CDP	TTL	CTYP	ASN	QUAL	SIR	CHRS
1 2 3 4 5 6 7 8 9	10 11 12 13 14 15 16 17 18 19 20 21 22 23 24	25 26	27 28 29 30	31 32 33 34	35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50	51 52	53 54	55 56	57 58 59 60 61 62 63 64 65	66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
									X X . X X X X . X	

955-1115-0

OUTPUT FORMAT FOR PGM9B

Logical Record Length: 250 Bytes

PL/1 Base Data Structure Name: RTRANS

Output PL/1 Filename: TRANS

Start Record Position	Byte Length	PL/1 Identifier	Format	Description
1-51	51	R51	A51	First 51 columns of input card (direct copy)
52-108	57	FILL	A57	Same on all output records; used to initialize the 14 time-type segments; see program listing, PL/1 Identifier = FILLER, for data content

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/I -PGM9A- FLOWCHARTS

PL/I MODULE (LIST,NAMSG,SOURCSEQ)

```
CARD NO   ****                          CONTENTS                                  ****

 1        OPTION PACKIF,COMPRESS
 2        PGM9A:  /* INSTR FILE TYPE -2- CARD TRANS PROCESSOR: SUN/4/20/75 */
 3                PROC OPTIONS (MAIN) ;
 4
 5        DECLARE CARD        CHAR (80) STATIC,
 6                CSI         CHAR (51) DEF CARD POS (1),
 7                CASN        CHAR (1) DEF CARD POS (53),
 8                COP         CHAR (4) DEF CARD POS (32),
 9                NC          FIXED BIN (15) STATIC INIT ( 0 ),
10                NCU         FIXED BIN (15) STATIC INIT ( 0 ),
11                RTRANS      CHAR (250) STATIC INIT ( ' ' ),
12                RSI         CHAR (51) DEF RTRANS POS (1),
13                FILL        CHAR (57) DEF RTRANS POS (52),
14                FILLER      CHAR (57) STATIC INIT
15        ( 'A1 A2 A3 A4 A5 A6 A7 B1 B2 B3 B4 B5 B6 B6A00.0000.00000.0' ),
16                M1          CHAR (56) STATIC INIT
17        ( '* TOTAL # INSTRUCTOR TYPE -2- CARDS PROCESSED = XXXX *' ),
18                M2          CHAR (51) STATIC INIT
19        ( '* TOTAL # INSTRUCTOR ASSIGNMENTS PROCESSED = XXXX *' ),
20                VM1         PICTURE 'ZZZ9' DEF M1 POS (49),
21                VM2         PICTURE 'ZZZ9' DEF M2 POS (46),
22                J           FIXED BIN (15) STATIC,
23                LBL         CHAR (2) STATIC INIT ( '* ' ) ;
24
25        DECLARE OLDID       CHAR (9) STATIC INIT ( ' ' ),
26                OLDCOP      CHAR (4) STATIC INIT ( ' ' ),
27                NDUP        FIXED BIN STATIC INIT ( 0 ),
28                NUNA        FIXED BIN STATIC INIT ( 0 ),
29                ID          CHAR (9) DEF CARD POS (1),
30                M3          CHAR (66) STATIC INIT (
31        '* UNASSIGNED RECORDS = X,XXX - DUPLICATE RECORDS SKIPPED = X,XXX' ),
32                VM3A        PICTURE 'Z,ZZ9' DEF M3 POS (24),
33                VM3B        PICTURE 'Z,ZZ9' DEF M3 POS (60) ;
34
35        STARTERRUP:
36                OPEN FILE (TRANS) RECORD OUTPUT,
37                     FILE (CARDS) RECORD INPUT ;
38                ON ENDFILE (CARDS) GO TO WRAPUP ;
39                FILL = FILLER ;
40
41        R1:     READ FILE (CARDS) INTO (CARD) ;
42                NC = NC + 1 ;
43
44                  IF ( CASN = '1' ) THEN GO TO TRYDUP ;
45        NOASN:          NUNA = NUNA + 1 ;
46                          GO TO R1 ;
47
48        TRYDUP: IF ( ID == OLDID ) THEN GO TO HERE_ASSIGNED ;
49                IF ( COP == OLDCOP ) THEN GO TO HERE_ASSIGNED ;
50                      NDUP = NDUP + 1 ;
51                          GO TO R1 ;
52
53        HERE_ASSIGNED:
54                OLDID = ID ;
55                OLDCOP = COP ;
56                NCU = NCU + 1 ;
57                RSI = CSI ;
58                WRITE FILE (TRANS) FROM (RTRANS) ;
59                     GO TO R1 ;
60
61        WRAPUP: CLOSE FILE (CARDS) ;
62                CLOSE FILE (TRANS) ;
63                VM1 = NC ;
64                VM2 = NCU ;
65                VM3A = NUNA ;
66                VM3B = NDUP ;
67                DISPLAY ( LBL ) ;
68                DISPLAY ( M1 ) ;
69                DISPLAY ( M2 ) ;
70                DISPLAY ( M3 ) ;
71
72                     END PGM9A ;
```

CHART TITLE - PROC PGM9B O[illegible]NS (MAIN)

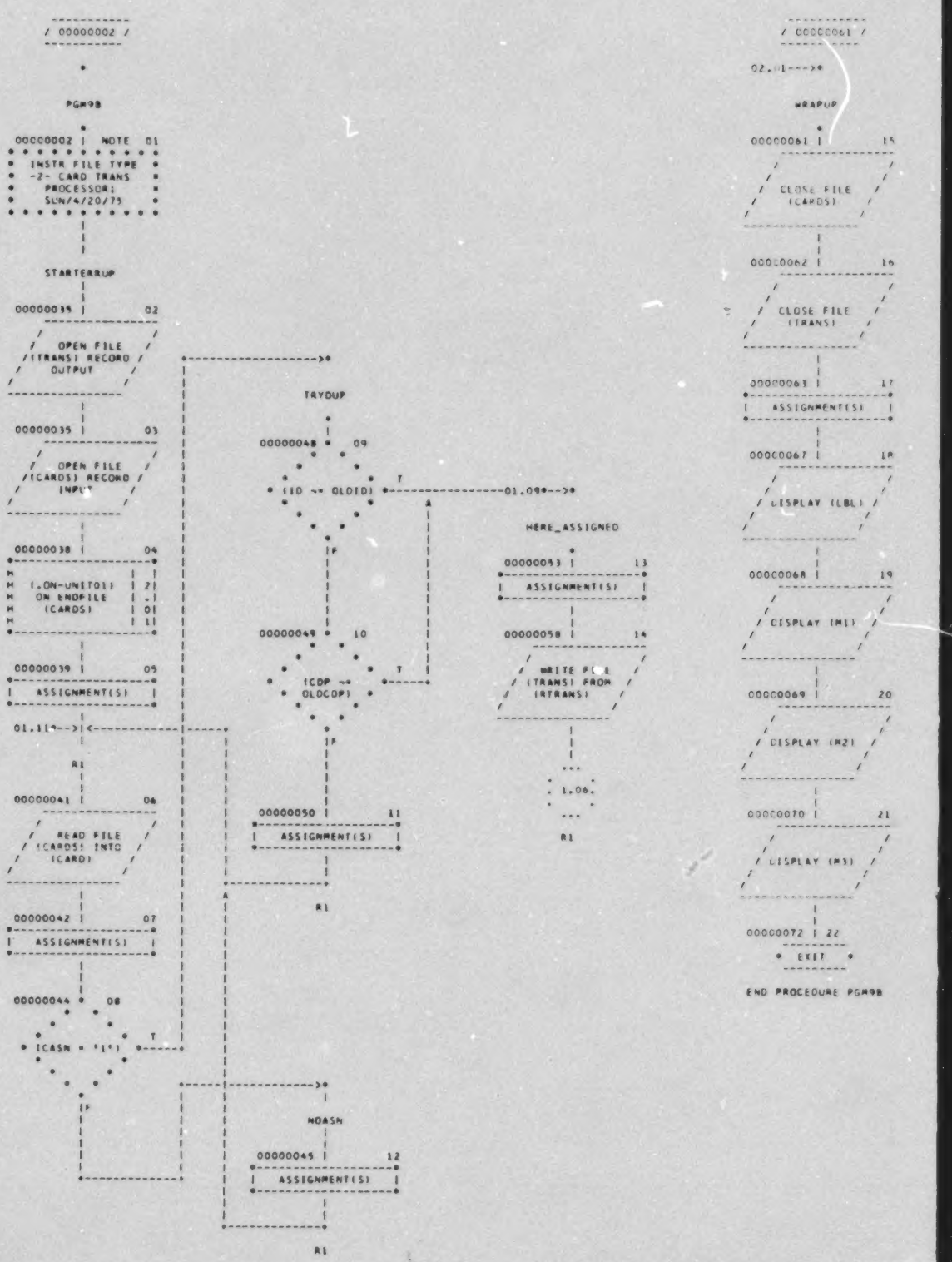


CHART TITLE - ON-UNIT ACTION BLOCKS

```
          -----------
         / 00000038 /
         -----------

 01.04--->*

.ON-UNIT01

              *
 00000038 |  NOTE  C
* * * * * * * * * * *
*    ON ENDFILE     *
*      (CARDS)      *
* * * * * * * * * * *
              |
              |
              |
             ...
           .     .
           . 1.15.
           .     .
             ...

          WRAPUP

     END .ON-UNIT01
```

PROGRAM NAME: PGM16

TYPE: PL/1

SYSTEM: Data Base Maintenance

NCSS FILENAME/FILETYPE: PGM16/PLI

A listing of the program source code and machine run flowcharts are included in this document on the pages following.

FUNCTION

This program combines the data found on Type -1- and Type -2- input cards into one logical record for initilization of the Course data base. One output record (transaction) is produced for each pair of input cards. Output transactions are subsequently processed into the Course data base by the RAMIS data maintenance INPUT function RE43/RAM (filename/filetype).

USAGE

Program PGM16 is invoked during operation of the following Root Executive Sequences (NCSS Filetype = EXEC):

EXEC Sequence	Data Base
E43	Course Master

INPUT FORMAT

Input data card formats are illustrated in the diagrams titled: "Course Data Base Type -1- Input Card Format" and "Course Data Base Type -2- Input Card Format" on the pages following. These cards must be stacked in the input stream in pairs; i.e., for a given Course, both types must be present. Type -1- precedes the Type -2- format card.

OUTPUT FORMAT FOR PGM16

Logical Record Length: 250 Bytes (218 Bytes contain data)
PL/1 Base Data Structure Name: RTRANS
Output PL/1 Filename: TRANS

LEVEL	FIELD NAME	POSITIONS IN RECORD	TYPE	LENGTH
1	SCH	1-4	A	4
2	CDP	5-8	A	4
3	DEL	9	A	1
	CTLG	10-19	A	10
	CRS	20-35	A	16
	CONV1	36-38	I	3
	LEN1	39-42	F	4.1
	CAP1	43-45	I	3
	BCAP1	46-48	I	3
	DMD1	49-53	I	5
	BDMD1	54-58	I	5
	OFST	59-60	I	2
	PFAIL	61-65	F	5.3
	NADIS	66-70	F	5.3
	NOSHOW	71-75	F	5.3
	SETB	76-80	F	5.3
	BKLG	81-84	I	4
	JNO	85-88	A	4
	CONV2	89-91	I	3
	LEN2	92-95	F	4.1
	CAP2	96-98	I	3
	BCAP2	99-101	I	3
	DMD2	102-106	I	5
	BDMD2	107-111	I	5
	CHWK2	112-114	I	3
	CONV3	115-117	I	3
	LEN3	118-121	F	4.1
	CAP3	122-124	I	3
	BCAP3	125-127	I	3
	DMD3	128-132	I	5
	BDMD3	133-137	I	5
	CHWK3	138-140	I	3
	PAOB	141-144	I	4
	AFAIL	145-149	F	5.3
	PRTY	150-151	I	2
	CTYPE	152-153	A	2
	PNEC	154-157	A	4
	UCAP	158-160	I	3
	UDEM	161-163	I	3
	FPRTY	164-165	I	2
	RDTE	166-170	I	6
	HSETB	171-176	F	5.3
	TINSTR	177-180	F	4.1
	SUIC	181-185	A	5
	CCMM	186-195	A	10

LEVEL	FIELD NAME	POSITIONS IN RECORD	TYPE	LENGTH
4	SIR	196-199	F	4.1
	CHRS	200-204	F	5.1
	CTYP	205	A	1
5	ID	206-214	A	9
	QUAL	215-217	A	3
	ASN	218	I	1

NOTE - RECORD POSITIONS 154-218, INCLUSIVE, ARE INITIALIZED ONE TIME AT PROGRAM PGM16 INITIALIZATION; NO DATA ARE TRANSFERRED INTO THESE POSITIONS FROM THE INPUT CARDS.

ALPHABETIC (TYPE = A) FIELDS ARE INITIALIZED TO BLANKS; NUMERIC FIELDS (TYPE = I OR F) ARE INITIALIZED TO ZERO.

COURSE DATA BASE TYPE -1- INPUT CARD FORMAT

80 Column Card Punch Layout

IBM Federal Systems Division, Space Systems Center, Huntsville, Alabama

NAME: COURSE CARD NO. 1

CDP	CTLG	CRS	CONV 1	LEN 1	CAP 1	BCAP 1	DMD1	BDMD1	OF ST	PFAIL	NADIS	NO SHOW	SETB	BKLG	JNO		
1	2 3 4	5 6 7 8 9 10 11 12 13 14	15 16 17 18 19 20 21 22 23 24 25 26 27 28 29	30 31 32 33	34 35 36	37 38	39 40 41 42 43 44 45 46 47	48 49 50 51 52	53 54	55 56 57	58 59 60 61 62	63 64 65	66 67 68 69	70 71 72 73	74 75 76 77	78 79	80
																	1
																	1

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

955-1115-0

COURSE DATA BASE TYPE -2- INPUT CARD FORMAT

IBM Federal Systems Division, Space Systems Center, Huntsville, Alabama

80-Column Card Punch Layout

NAME: COURSE CARD NO. 2

JOB NO

WORK PACKAGE

WRITTEN AS

PUNCH AS

CARD PUNCH PRINT YES NO

VERIFY YES NO

DATE DUE OUT

SHEET 2 OF 2

CDP	CONV 2	LEN 2	CAP 2	BCAP 2	DMD2	BDMD2	CH WK2	CONV 3	LEN 3	CAP 3	BCAP 3	DMD3	BDMD3	CH WK3	PAOS	AFAIL	PRTY	CTYPE	PNEC	SCH	SUIC	
1 2 3 4	5 6 7	8 9 10	11 12 13	14 15 16	17 18 19 20	21 22 23 24 25 26	27 28 29	30 31 32	33 34 35	36 37 38	39 40 41	42 43 44 45 46	47 48 49 50	51 52 53	54 55 56	57 58 59 60 61	62 63	64 65	66 67 68 69 70	71 72 73	74 75 76 77 78	79 80
																						2 2

955-1115-0

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PGM16- FLOWCHARTS

PL/I MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
PGM16:  /* COURSE FILE TYPES 1/2 CARD COMBINE - THU/24/APR/75 */
        PROC OPTIONS (MAIN) ;

DECLARE ( CARD1, CARD2 ) CHAR (80) STATIC,
        CCDP1     CHAR (4) DEF CARD1 POS (1),
        TCRD1     CHAR (1)  DEF CARD1 POS (80),
        F1        CHAR (29) DEF CARD1 POS (5),
        F2        CHAR (18) DEF CARD1 POS (37),
       F3        CHAR (8)   DEF CARD1 POS (71),
        CCDP2     CHAR (4)  DEF CARD2 POS (1),
        CSCH      CHAR (4)  DEF CARD2 POS (71),
        TCRD2     CHAR (1)  DEF CARD2 POS (80),
        F4        CHAR (3)  DEF CARD2 POS (5),
        F5        CHAR (22) DEF CARD2 POS (11),
        F6        CHAR (23) DEF CARD2 POS (36),
        F7        CHAR (8)  DEF CARD2 POS (63),
        F8        CHAR (5)  DEF CARD2 POS (75) ;

DECLARE RTRANS    CHAR (250) STATIC INIT ( ' ' ), /* 218 CHARS USED */
        TSCH      CHAR (4) DEF RTRANS POS (1),
        TCDP      CHAR (4) DEF RTRANS POS (5),
        TDEL      CHAR (1) DEF RTRANS POS (9),
        F1A       CHAR (29) DEF RTRANS POS (10),
        F2A       CHAR (18) DEF RTRANS POS (43),
        F3A       CHAR (8)  DEF RTRANS POS (81),
        LEN1      PICTURE '99V.9'     DEF RTRANS POS (39),
        PFAIL     PICTURE '9V.999'    DEF RTRANS POS (61),
        NADIS     PICTURE '9V.999'    DEF RTRANS POS (66),
        NOSHOW    PICTURE '9V.999'    DEF RTRANS POS (71),
        SETB      PICTURE '9V.999'    DEF RTRANS POS (76),
        F4A       CHAR (3)       DEF RTRANS POS (89),
        F5A       CHAR (22)      DEF RTRANS POS (96),
        F6A       CHAR (23)      DEF RTRANS POS (122),
        F7A       CHAR (8)       DEF RTRANS POS (150),
        F8A       CHAR (5)       DEF RTRANS POS (181),
        LEN2      PICTURE '99V.9'     DEF RTRANS POS (92),
        LEN3      PICTURE '99V.9'     DEF RTRANS POS (118),
        AFAIL     PICTURE '9V.999' DEF RTRANS POS (145),
        FILL1     CHAR (45)      DEF RTRANS POS (154),
        FILL2     CHAR (23)      DEF RTRANS POS (196) ;

DECLARE M1        CHAR (49) STATIC INIT (
'** ERROR # XX OCCURED: INPUT CARD COUNT = XXXX **' ),
        E#        PICTURE 'Z9' DEF M1 POS (12),
        NC#       PICTURE 'ZZZ9' DEF M1 POS (43),
        C265      CHAR (65) DEF CARD2 POS (1),
        C165      CHAR (65) DEF CARD1 POS (1),
        M2        CHAR (60) STATIC INIT (
'** PROGRAM PGM16 TERMINATING - XXXX INPUT CARDS PROCESSED **' ),
        NC#2      PICTURE 'ZZZ9' DEF M2 POS (32),
        NER       FIXED BIN STATIC,
        NC        FIXED BIN STATIC INIT ( 0 ) ;

STARTERRUPP:
        OPEN FILE (CARDS) RECORD INPUT,
             FILE (TRANS) RECORD OUTPUT ;
        ON ENDFILE (CARDS) GO TO WRAPUP ;
        TDEL = '1' ;
        FILL1 = '    000000000000000000.00000.0             ' ;
        FILL2 = '00.0000.0             1' ;

R1:     NER = 1 ;
        READ FILE (CARDS) INTO (CARD1) ;
             NC = NC + 1 ;
        IF ( TCRD1 ¬= '1' ) THEN GO TO BUST ;
ER2:    NER = 2 ;
        READ FILE (CARDS) INTO (CARD2 ) ;
             NC = NC + 1 ;
        IF ( TCRD2 ¬= '2' ) THEN GO TO BUST ;
ER3:    NER = 3 ;
        IF ( CCDP1 ¬= CCDP2 ) THEN GO TO BUST ;

ER4:    NER = 4 ;
DO_CARD1:
        TCDP = CCDP1 ;
        TSCH = CSCH ;
        F1A = F1 ;
        F2A = F2 ;
        F3A = F3 ;
        GET STRING (CARD1) EDIT
             ( LEN1, PFAIL, NADIS, NOSHOW, SETB )
             ( X(33), F(3,1), X(18), 4 (F(4,3)) ) ;
DO_CARD2:
        F4A = F4 ;
        F5A = F5 ;
        F6A = F6 ;
        F7A = F7 ;
        F8A = F8 ;
        GET STRING (CARD2) EDIT
             ( LEN2, LEN3, AFAIL )
             ( X(7), F(3,1), X(22), F(3,1), X(23), F(4,3) ) ;
        WRITE FILE (TRANS) FROM (RTRANS) ;
                GO TO R1 ;

BUST:   E# = NER ;
        NC# = NC ;
             DISPLAY ( M1 ) ;
             DISPLAY ( '** CARD1 **' || C165 ) ;
             DISPLAY ( '** CARD2 **' || C265 ) ;
                GO TO FINIS ;

WRAPUP:
        NC#2 = NC ;
             DISPLAY ( M2 ) ;
FINIS:  CLOSE FILE (CARDS),
              FILE (TRANS) ;

        END PGM16 ;
```

CHART TITLE - PROC PGM16 OPTIONS (MAIN)

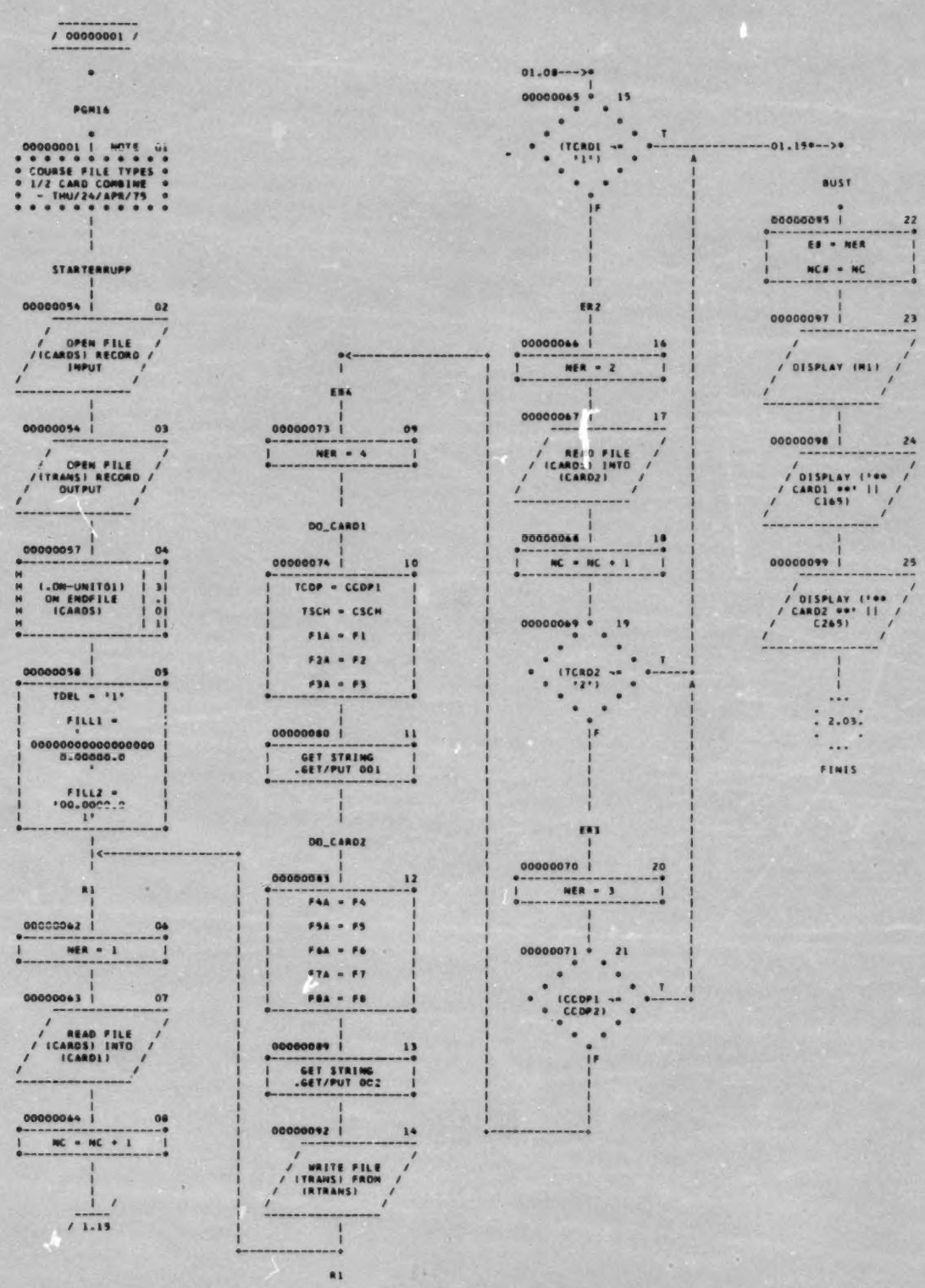


CHART TITLE - PROC PGM16 OPTIONS (MAIN)

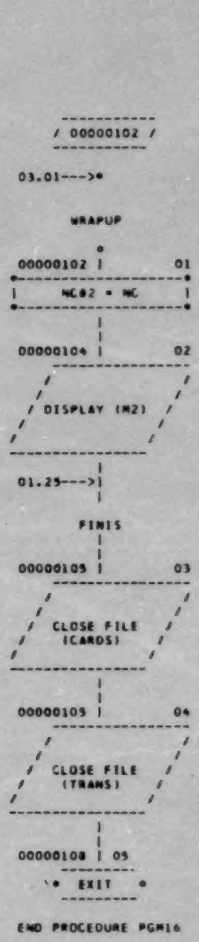


CHART TITLE - ON-UNIT ACTION BLOCKS

```
          -----------
         / 00000057 /
         -----------

 01.04--->*

.ON-UNIT01

                 *
 00000057 |  NOTE  01
* * * * * * * * * * *
*    ON ENDFILE     *
*      (CARDS)      *
* * * * * * * * * * *
          |
          |
          |
         ...
        .     .
        . 2.01.
        .     .
         ...

        WRAPUP

    END .ON-UNIT01
```

226

CHART TITLE - GET/PUT, FORMAT STATEMENTS

```
00000080          .GET/PUT 001  GET STRING (CARD1) EDIT (LEN1,PFAIL,NADIS,NOSHOW,SETB) (X (33),F (3,1),X (18),
00000082                        [illegible] (4,3)))
00000089          .GET/PUT 002  GET STRING (CARD2) EDIT (LEN2,LEN3,AFAIL) (X (7),F (3,1),X (22),F (3,1),X (23),F (4,
00000091                        3))
```

227

PROGRAM NAME: PGM19

TYPE: PL/1

SYSTEM: Data Base Maintenance

NCSS FILENAME/FILETYPE: PGM19/PLI

A listing of the program source code and machine run flowcharts are included in this document on the pages following.

FUNCTION

Following a standard RAMIS data maintenance DUMP of the Course data base, PGM19 is used to display each of the RAMIS records at the User terminal.

USAGE

Program PGM19 is invoked during operation of the following Root Executive Sequences (NCSS Filetype = EXEC):

EXEC Sequence	Data Base
E58	Course Master
E37	Course Scratch

INPUT FORMAT

For the Course data base, the field positions in the output of the RAMIS DUMP are indicated in the following RAMIS sample output. Note that the DUMP file takes the form of fixed-length card-image records (logical record length = 80 Bytes).

‡ EBCDIC RECORD LENGTH 8Ø CHARACTERS

FIELD NAME	FORMAT	LENGTH
SCH	A 4	4
CDP	A 4	4
DEL	A 1	1
CTLG	A 1Ø	1Ø
CRS	A 16	16
CONV1	I 3	3
LEN1	F4 .1	4
CAP1	I 3	3
BCAP1	I 3	3
DMD1	I 5	5
BDMD1	I 5	5
OFST	I 2	2
PFAIL	F5 .3	5

FIELD NAME	FORMAT	LENGTH
NADIS	F5 .3	5
NOSHOW	F5 .3	5
SETB	F5 .3	5
		X Ø
BKLG	I 4	4
JNO	A 4	4
CONV2	I 3	3
LEN2	F4 .1	4
CAP2	I 3	3
BCAP2	I 3	3
DMD2	I 5	5
BDMD2	I 5	5
CHWK2	I 3	3
CONV3	I 3	3
LEN3	F4 .1	4
CAP3	I 3	3
BCAP3	I 3	3
DMD3	I 5	5
BDMD3	I 5	5
CHWK3	I 3	3
PAOB	I 4	4
AFAIL	F5 .3	5
PRTY	I 2	2
CTYPE	A 2	2
PNEC	A 4	4
UCAP	I 3	3
		X Ø
UDEM	I 3	3
EPRTY	I 2	2
RDTE	I 6	6
HSETB	F5 .3	5
TINSTR	F4 .1	4
SUIC	A 5	5
CCMM	A 1Ø	1Ø
SIR	F4 .1	4
CHRS	F5 .1	5
CTYP	A 1	1
ID	A 9	9
QUAL	A 3	3
ASN	I 1	1

Zeros at the positions marked by 'X' indicate that no slack bytes were required at the end of each card image to prevent split of a data field across card boundaries. Three (3) card-image records are required in the DUMP to represent one RAMIS record; a portion of the last card image of the trio remains unused.

OUTPUT RECORDS

Output of PGM 19 is directed to the User terminal. Three (3) input card images are required (minimum) in the construction of an output terminal screen display.

Output is composed to fit on the 80-Character display face; essentially, an entire screen is required to display the data associated with a single Course.

A sample of the terminal output is illustrated by the diagram following.

SAMPLE TERMINAL OUTPUT PRODUCED BY PROGRAM PGM19

```
2573      BW/FW T/T RECERT      --SCHEDULE DATA--   FY   FY+1   FY+2   CHANGE DATE
                                ANNUAL CONVENINGS   11     Ø      Ø   OFFSET     Ø
SCHL JNO  CATALOG NO  PNEC      COURSE LENGTH      3.Ø    .Ø    .Ø   FY+1 WK    Ø
AC&R      A-651-ØØ51            COURSE CAPACITY     16     Ø      Ø   FY+2 WK    Ø
                                BUPERS CAPACITY      Ø     Ø      Ø
PAOB PFAIL AFAIL TYPE PRTY      ANNUAL DEMAND       46    46     46   BACKLOG    Ø
   Ø  .ØØØ  .ØØØ  F1    Ø      BUPERS DEMAND        Ø     Ø      Ø

   CCMM      SUIC     RDTE     FPRTY  NADIS  NOSHOW SETB  ASETB  UCAP UDEM  TINSTR
                        Ø       Ø     .ØØØ   .ØØØ    .ØØØ  .ØØØ    26    Ø     .Ø

                    !------------------ INSTRUCTOR ASSIGNMENTS ------------------
SIR   CHRS   CTYP   !INSTR-ID  QUAL ASSN!INSTR-ID  QUAL ASSN!INSTR-ID  QUAL  ASSN
  .Ø     .Ø         !                 1 !                  !
                    !                   !                  !
18.Ø    5.Ø  1      !123456789  25   1 !222222222  1ØØ  1 !333333333  1ØØ   1
                    !455556667  Ø    Ø
```

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PGM19- FLOWCHARTS

PL/1 MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
 1   OPTION PACKIF,COMPRESS
 2   PGM19:  /* COURSE FILE PUMP TO DISPLAY - THRU/B/MAY/75 */
 3           PROC OPTIONS (MAIN) ;
 4
 5   DECLARE SICLR      CHAR (190) STATIC,
 6           LBLANK     CHAR (5) STATIC INIT ( ' ' ),
 7           SIROW (10) CHAR (19) DEF SICLR POS (1),
 8           SIBLD      CHAR (19) STATIC INIT ( ' ' ),
 9                BSIR  CHAR (4) DEF SIBLD POS (1),
10                BCHRS CHAR (5) DEF SIBLD POS (7),
11                BCTYP CHAR (4) DEF SIBLD POS (14),
12           OLOSIR     CHAR (4) STATIC INIT ( ' ' ) ;
13
14   DECLARE INCLR      CHAR (600) STATIC,
15           INROW (10) CHAR (60) DEF INCLR POS (1),
16           INTAB (10,3) CHAR (20) DEF INCLR POS (1),
17           INBLD      CHAR (20) STATIC INIT ( '|' ),
18                BID   CHAR (9) DEF INBLD POS (2),
19                BQUAL CHAR (3) DEF INBLD POS (13),
20                BASSN CHAR (3) DEF INBLD POS (18) ;
21
22   DECLARE DLINE      CHAR (79) STATIC,
23           L1         CHAR (79) STATIC INIT (
24   '                                --SCHEDULE DATA--    FY   FY+1   FY+2   CH0
25   ANGE DATE' ),
26           L2         CHAR (79) STATIC INIT (
27   '                                 ANNUAL CONVENINGS  XXX    XXX    XXX   OF0
28   FSET   XX' ),
29                L2CON1     CHAR (3) DEF L2 POS (49),
30                L2CON2     CHAR (3) DEF L2 POS (56),
31                L2CON3     CHAR (3) DEF L2 POS (63),
32                L2OFF      CHAR (2) DEF L2 POS (78),
33           L3         CHAR (79) STATIC INIT (
34   'SCHL JNO  CATALOG NO  PNEC   COURSE LENGTH      XX.X   XX.X   XX.X   FY0
35   +1 WK XXX' ),
36                L3LEN1     CHAR (4) DEF L3 POS (48),
37                L3LEN2     CHAR (4) DEF L3 POS (55),
38                L3LEN3     CHAR (4) DEF L3 POS (62),
39                L3WK1      CHAR (3) DEF L3 POS (77),
40                L1CDP      CHAR (4)  DEF L1 POS (1),
41                L1CRS      CHAR (16) DEF L1 POS (10),
42           L4         CHAR (79) STATIC INIT (
43   '                                 COURSE CAPACITY    XXX    XXX    XXX   FY0
44   +2 WK XXX' ),
45                L4SCH      CHAR (4)  DEF L4 POS (1),
46                L4JNO      CHAR (4)  DEF L4 POS (6),
47                L4CAT      CHAR (10) DEF L4 POS (11),
48                L4PNEC     CHAR (4)  DEF L4 POS (23),
49                L4CAP1     CHAR (3)  DEF L4 POS (49),
50                L4CAP2     CHAR (3)  DEF L4 POS (56),
51                L4CAP3     CHAR (3)  DEF L4 POS (63),
52                L4WK2      CHAR (3)  DEF L4 POS (77),
53           L5         CHAR (79) STATIC INIT (
54   '                                 BUPERS CAPACITY' ),
55                L5CAP1     CHAR (3) DEF L5 POS (49),
56                L5CAP2     CHAR (3) DEF L5 POS (56),
57                L5CAP3     CHAR (3) DEF L5 POS (63),
58           L6         CHAR (79) STATIC INIT (
59   'PACB PFAIL AFAIL TYPE PRTY   ANNUAL DEMAND    XXXXX  XXXXX  XXXXX  BAC
60   KLOG XXXX' ),
61                L6DMD1     CHAR (5) DEF L6 POS (47),
62                L6DMD2     CHAR (5) DEF L6 POS (54),
63                L6DMD3     CHAR (5) DEF L6 POS (61),
64                L6BKLOG    CHAR (4) DEF L6 POS (76),
65           L7         CHAR (79) STATIC INIT (
66   '                                 BUPERS DEMAND    XXXXX  XXXXX  XXXXX' ),
67                L7ACB      CHAR (4) DEF L7 POS (1),
68                L7PF       CHAR (5) DEF L7 POS (6),
69                L7AF       CHAR (5) DEF L7 POS (12),
70                L7TYPE     CHAR (2) DEF L7 POS (19),
71                L7PRTY     CHAR (2) DEF L7 POS (24),
72                L7BD1      CHAR (5) DEF L7 POS (47),
73                L7BD2      CHAR (5) DEF L7 POS (54),
74                L7BD3      CHAR (5) DEF L7 POS (61),
75           L9         CHAR (79) STATIC INIT (
76   '   CCMM      SUIC     RDTE   FPRTY  NADIS  NOSHOW SETB  ASETB  UCAP UDEM
77     TINSTR' ),
78           L10        CHAR (79) STATIC INIT ( ' ' ),
79                L10CCMM    CHAR (10)DEF L10 POS (1),
80                L10SUIC    CHAR (5) DEF L10 POS (13),
81                L10RDTE    CHAR (6) DEF L10 POS (20),
82                L10PRTY    CHAR (2) DEF L10 POS (29),
83                L10NADIS   CHAR (5) DEF L10 POS (35),
84                L10NOSHOW  CHAR (5) DEF L10 POS (42),
85                L10SETB    CHAR (5) DEF L10 POS (50),
86                L10ASETB   CHAR (5) DEF L10 POS (56),
87                L10UCAP    CHAR (3) DEF L10 POS (64),
88                L10UDEM    CHAR (3) DEF L10 POS (69),
89                L10TIN     CHAR (4) DEF L10 POS (75),
90           L12        CHAR (79) STATIC INIT (
91   '                    |---------------- INSTRUCTOR ASSIGNMENTS ----------
92   ---------' ),
93           L13        CHAR (79) STATIC INIT (
94   'SIR   CHRS   CTYP  |INSTR-ID  QUAL ASSN|INSTR-ID  QUAL ASSN|INSTR-ID
95   QUAL ASSN' ),
96           LBL2       CHAR (20) STATIC INIT ( '|' ) ;
97
98   DECLARE 1 CARD1   STATIC,
99             2 SCH    CHAR (4),
100            2 CDP    CHAR (4),
101            2 DEL    CHAR (1),
102            2 CTLG   CHAR (10),
103            2 CRS    CHAR (16),
104            2 CONV1  CHAR (3),
105            2 LEN1   CHAR (4),
106            2 CAP1   CHAR (3),
107            2 BCAP1  CHAR (3),
108            2 DMD1   CHAR (5),
109            2 BDMD1  CHAR (5),
```

232

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PGM15- FLOWCHARTS

CARD NO **** CONTENTS ****

```
                   2 OFST    CHAR (2),
                   2 PFAIL   CHAR (5),
                   2 NADIS   CHAR (5),
                   2 NOSHOW  CHAR (5),
                   2 SETB    CHAR (5) ;

      DECLARE 1 CARD2    STATIC,
                   2 BKLG    CHAR (4),
                   2 JNO     CHAR (4),
                   2 CONV2   CHAR (3),
                   2 LEN2    CHAR (4),
                   2 CAP2    CHAR (3),
                   2 BCAP2   CHAR (3),
                   2 DMD2    CHAR (5),
                   2 BDMD2   CHAR (5),
                   2 CHWK2   CHAR (3),
                   2 CONV3   CHAR (3),
                   2 LEN3    CHAR (4),
                   2 CAP3    CHAR (3),
                   2 BCAP3   CHAR (3),
                   2 DMD3    CHAR (5),
                   2 BDMD3   CHAR (5),
                   2 CHWK3   CHAR (3),
                   2 PADB    CHAR (4),
                   2 AFAIL   CHAR (5),
                   2 PRTY    CHAR (2),
                   2 CTYPE   CHAR (2),
                   2 PNEC    CHAR (4),
                   2 UCAP    CHAR (3) ;

      DECLARE 1 CARD3     STATIC,
                   2 UDEM    CHAR (3),
                   2 FPRTY   CHAR (2),
                   2 ROTE    CHAR (6),
                   2 HSETB   CHAR (5),
                   2 TINSTR  CHAR (4),
                   2 SUIC    CHAR (5),
                   2 CCMM    CHAR (10),
                   2 SIR     CHAR (4),
                   2 CHRS    CHAR (5),
                   2 CTYP    CHAR (1),
                   2 ID      CHAR (9),
                   2 QUAL    CHAR (3),
                   2 ASN     CHAR (1),
                   2 GARBG CHAR (22) ;

      DECLARE BEOF        BIT (1) STATIC INIT ( '0'B ),
              NC          FIXED BIN STATIC INIT ( 0 ),
              SOLDCOP     CHAR (4) STATIC INIT ( ' ' ),
              NL          FIXED BIN STATIC INIT ( 0 ),
              N           FIXED BIN STATIC INIT ( 0 ),
              NCRS        FIXED BIN STATIC INIT ( 0 ),
              M1          CHAR (55) STATIC INIT (
      '** ECF SEQUENCING ERROR IN PGM16 - CARD COUNT IS XX,XXX' ),
                    FILLM1      PICTURE 'ZZ,ZZ9' DEF M1 POS (50),
              M2          CHAR (34) STATIC INIT (
      '** TOTAL COURSES PROCESSED = X,XXX' ),
                    FILLM2      PICTURE 'Z,ZZ9' DEF M2 POS (30),
              J           STATIC,
              ROUTE       LABEL ;


      STARTERRUPP:
              OPEN FILE (DUMP) RECORD INPUT ;
              ON ENDFILE (DUMP) GO TO WINDUP ;

      READ1:  READ FILE (DUMP) INTO (CARD1) ;
                    BEOF = '1'B ;
              READ FILE (DUMP) INTO (CARD2) ;
              READ FILE (DUMP) INTO (CARD3) ;
                    BEOF = '0'B ;
              NC = NC + 1 ;

              IF ( COP ¬= SOLDCOP ) THEN
      NEWCOP:       DO ;
                    IF ( NL < 1 ) THEN GO TO START_NEW_COURSE ;
                          ROUTE = START_NEW_COURSE ;
                          GO TO DISPLAY_ONE_PAGE ;
                    END NEWCOP ;

       OLDCOP: IF ( SIR = OLDSIR ) THEN GO TO  SAMESIR ;
       NEWSIR: NL = NL + 1 ;
               IF ( NL > 10 ) THEN GO TO OUTNOWA ;
                     ROUTE = SIR2 ;
                     GO TO SETLINE ;
      SIR2:   IF ( NL = 10 ) THEN GO TO OUTNOW ;
                    GO TO RECYCLE1 ;

      /* HERE, OLD COP, SAME OLD SIR - MUST BE ANOTHER INSTRUCTOR */
       SAMESIR: N = N + 1 ;
              IF ( N <= 3 ) THEN GO TO FILL_INSTRUCTOR ;
              N = 1 ;
              NL = NL + 1 ;
              IF ( NL <= 10 ) THEN GO TO FILL_INSTRUCTOR ;
      OUTNOWA:      NL = 10 ;
      OUTNOW:       ROUTE = START_WHOLE_ASSIGNMENTS ;
                    GO TO DISPLAY_ONE_PAGE ;
      START_NEW_COURSE:
              NCRS = NCRS + 1 ;
              SOLDCOP = COP ;

      FILL_TOPPAGE:
              L1COP = COP :  L2CON1 = CONV1 :  L3LEN1 = LEN1 ;
              L1CRS = CRS :  L2CON2 = CONV2 :  L3LEN2 = LEN2 ;
                             L2CON3 = CONV3 :  L3LEN3 = LEN3 ;
                             L2OFF = OFST :    L3WK1 = CHWK2 ;

              L4SCH  = SCH ;     L5CAP1 = BCAP1 :  L7AOB   = PAOB ;
              L4JNO  = JNO ;     L5CAP2 = BCAP2 ;  L7PF    = PFAIL ;
              L4CAT  = CTLG ;    L5CAP3 = BCAP3 :  L7AF    = AFAIL ;
              L4PNEC = ^NEC ;                      L7TYPE  = CTYPE ;
              L4CAP1 = CAP; ;    L6DMD1  = DMD1 ;  L7PRTY  = PRTY ;
```

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PGM19- FLOWCHARTS

CARD NO **** CONTENTS ****

```
                L4CAP2 = CAP2 ;    L6DMD2  = DMD2 ;   L7BD1  = BDMD1 ;
                L4CAP3 = CAP3 ;    L6DMD3  = DMD3 ;   L7BD2  = BDMD2 ;
                L4WK2  = CHWK3 ;   L6BKLOG = BKLG ;   L7BD3  = BDMD3 ;

                L10CCMM  = CCMM ;    L10NADIS  = NADIS ;    L10UCAP = UCAP ;
                L10SUIC  = SUIC ;    L10NOSHOW = NOSHOW ;   L10UDEM = UDEM ;
                L10RDTE  = RDTE ;    L10SETB   = SETB ;     L10TIN  = TINSTR ;
                L10FPRTY = FPRTY ;   L10HSETB  = HSETB ;

        START_WHOLE_ASSIGNMENTS:
                NL = 0 ;
                SICLR = ' ' ;
                INCLR = ' ' ;
        RECYCLE1:
                N = 1 ;
                NL = NL + 1 ;
                        ROUTE = FILL_SI_RATIO ;
                                GO TO SETLINE ;

        FILL_SI_RATIO:
                BSIR = SIR ;
                BCHRS = CHRS ;
                BCTYP = CTYP ;
                        SIROW (NL) = SIBLD ;
                OLDSIR = SIR ;

        FILL_INSTRUCTOR:
                BID   = ID ;
                BQUAL = QUAL ;
                BASSN = ASN ;
                        INTAB ( NL, N ) = INBLD ;
                                GO TO READ1 ;


        DISPLAY_ONE_PAGE:
                DISPLAY ( LBLANK ) ;      /* PAGE CLEAR GOES HERE,        */
                DISPLAY ( LBLANK ) ;      /*     WHENEVER AVAILABLE...    */

        /* LINES 1 THRU 10, HERE */
                DISPLAY ( L1 ) ;   DISPLAY ( L2 ) ;   DISPLAY ( L3 ) ;
                DISPLAY ( L4 ) ;   DISPLAY ( L5 ) ;   DISPLAY ( L6 ) ;
                DISPLAY ( L7 ) ;   DISPLAY ( LBLANK ) ;
                DISPLAY ( L9 ) ;   DISPLAY ( L10 ) ;

        /* TITLES ( LINES 11-13 ), HERE */
                DISPLAY ( LBLANK ) ;
                DISPLAY ( L12 ) ;
                DISPLAY ( L13 ) ;

        /* NOW, FOR INSTRUCTOR ASSIGNMENTS SECTION */
        INOUT:        DO J = 1 TO NL ;
                       DLINE = SIROW(J) || INROW(J) ;
                      DISPLAY ( DLINE ) ;
                      END INOUT ;
                                GO TO ROUTE ;        /* VARIABLE EXIT ROUTER */


        SETLINE: /* SUBROUTINE TO INSERT BLANK VERTICAL SEPARATORS */
                INTAB (NL,1) = LBL2 ;
                INTAB (NL,2) = LBL2 ;
                INTAB (NL,3) = LBL2 ;
                        GO TO ROUTE ;             /* VARIABLE EXIT ROUTER */


        WINDUP:
                 ROUTE = LAST_STEP ;
                        GO TO DISPLAY_ONE_PAGE ;

         LAST_STEP:
                IF ( BEOF ) THEN
        BUMEOF:       DO ;
                        FILLM1 = NC ;
                                DISPLAY ( LBLANK ) ;
                                DISPLAY ( M1 ) ;
                        END BUMEOF ;

                FILLM2 = NCRS ;
                DISPLAY ( LBLANK ) ;
                DISPLAY ( M2 ) ;
                IF ( BEOF )
                        THEN DISPLAY ( '** PGM16 TERMINATING ABNORMALLY' ) ;

        CLOSEM: CLOSE [illegible] (DUMP) ;
                        END [illegible]19 ;
```

CHART TITLE - PROC PGM19 OPTIONS (MAIN)

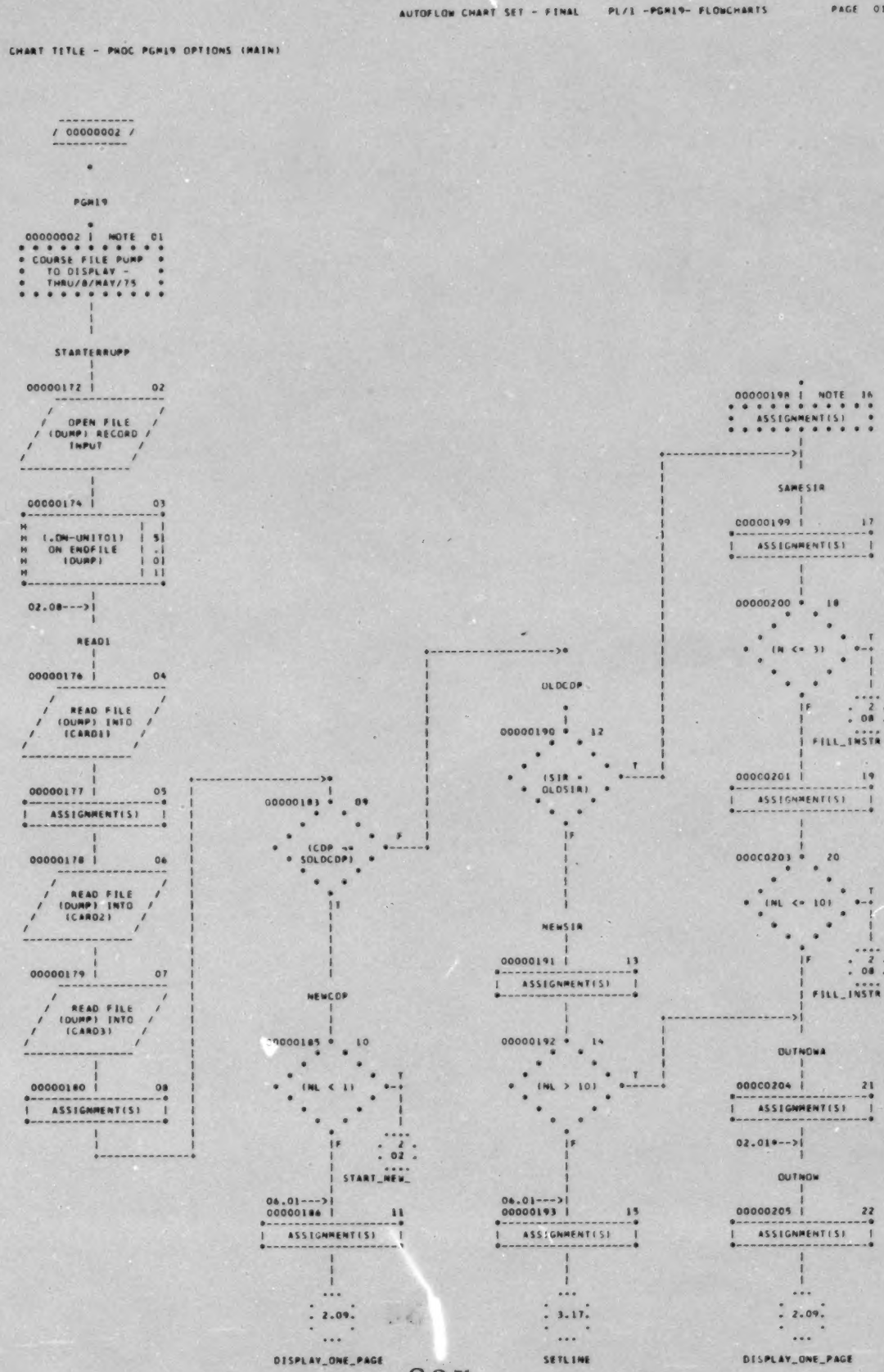


CHART TITLE - PROC PGM19 OPTIONS (MAIN)

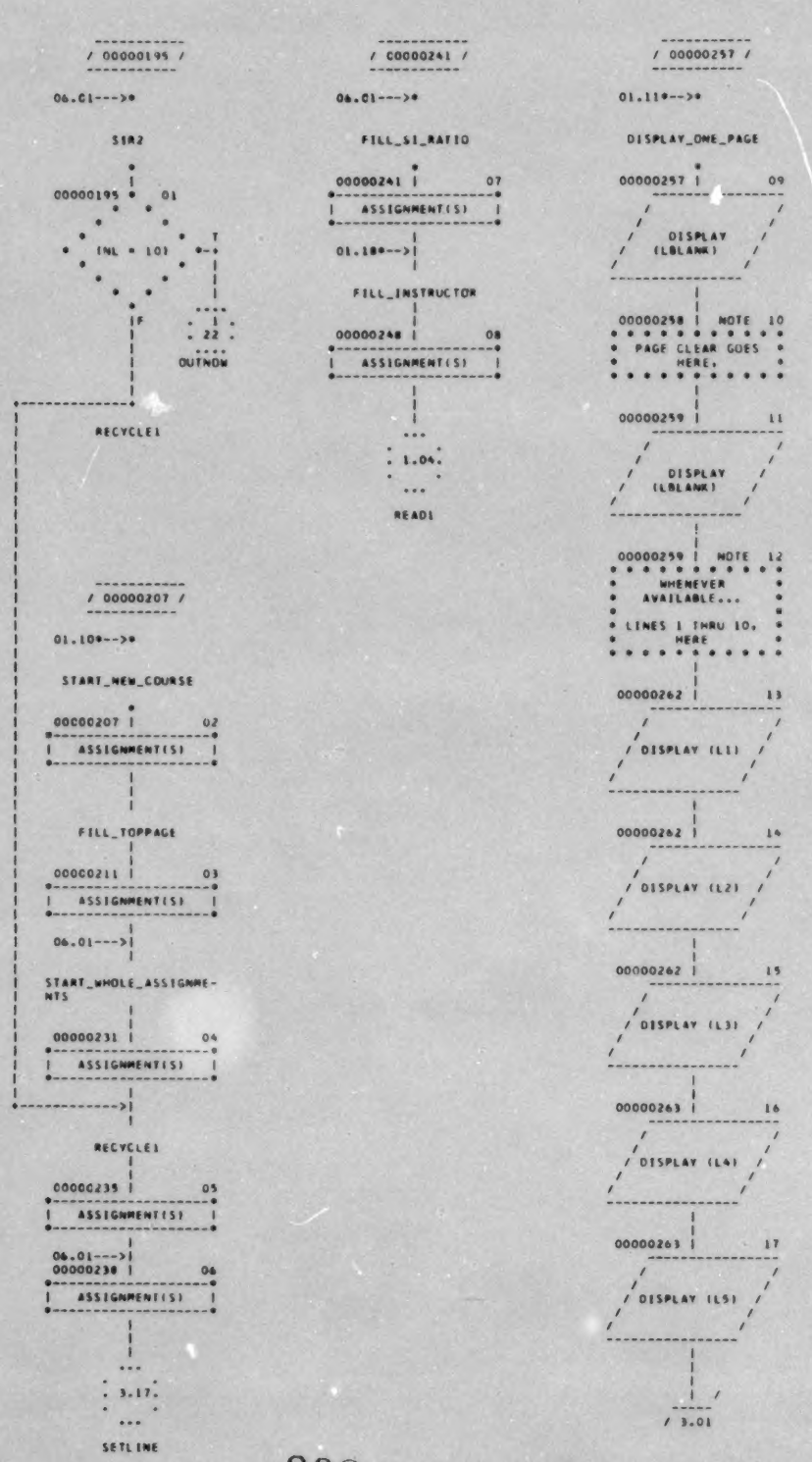


CHART TITLE - PROC PGM19 OPTIONS (MAIN)

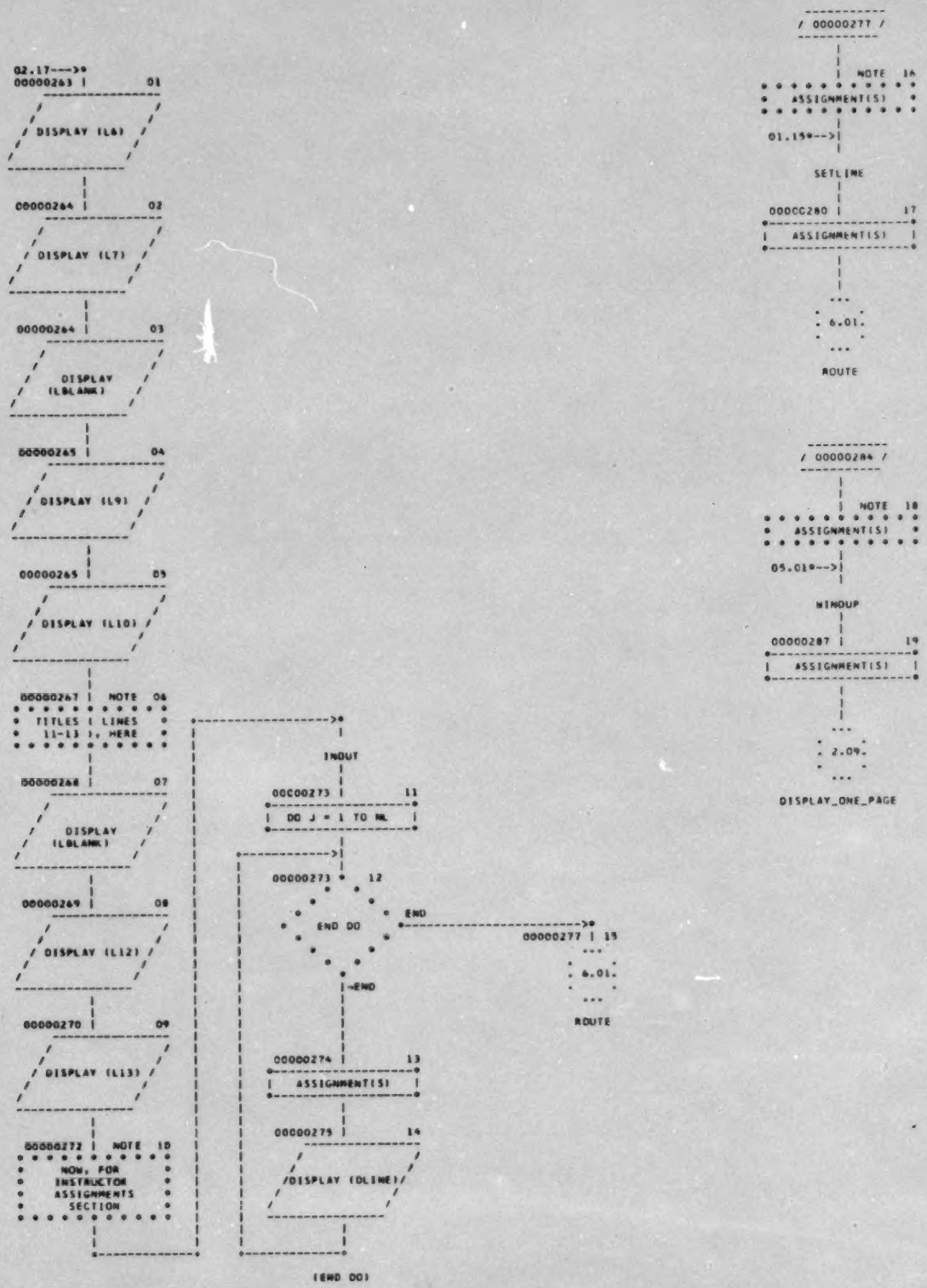


CHART TITLE - PROC PGM19 OPTIONS (MAIN)

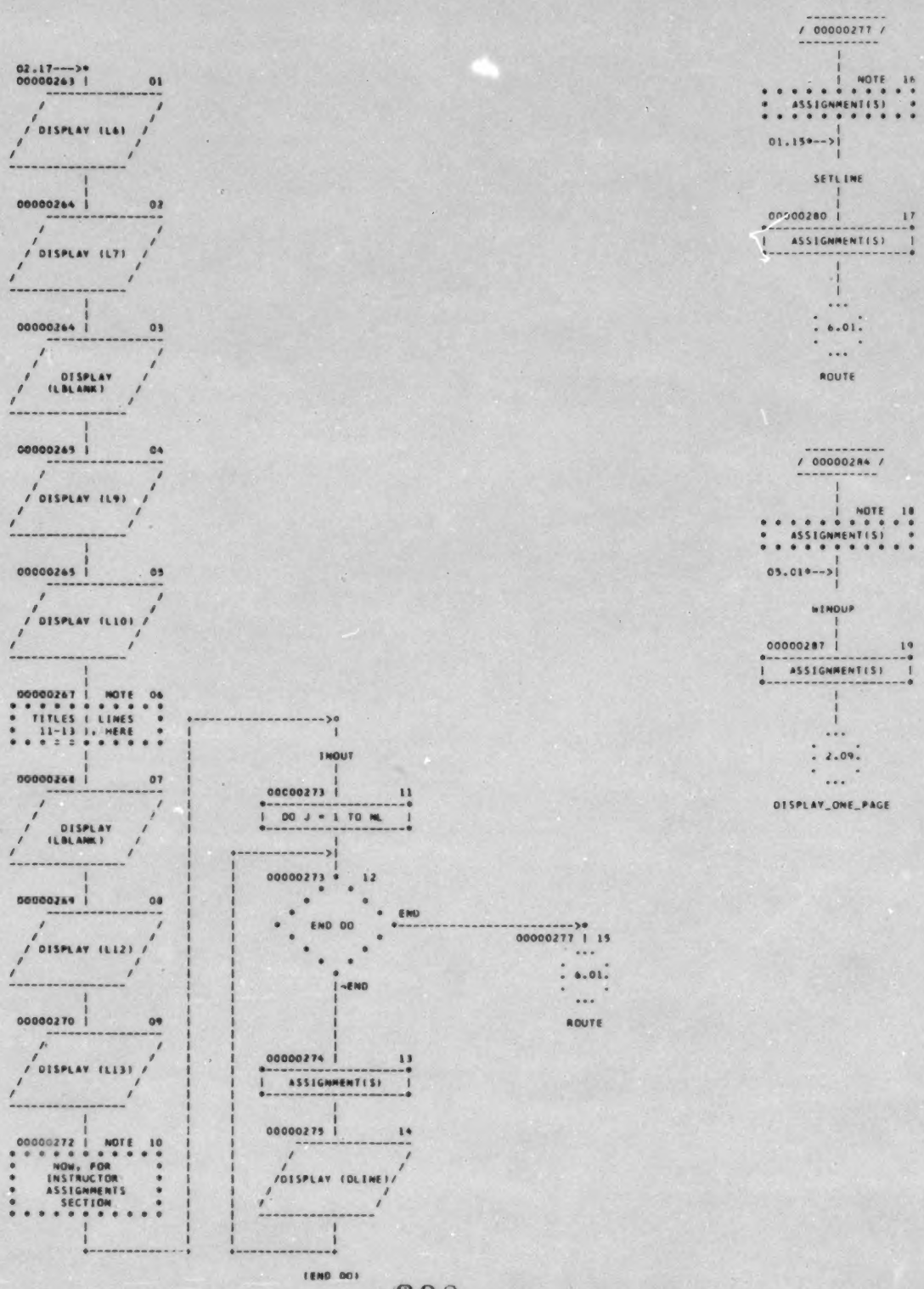


CHART TITLE - PROC PGM19 OPTIONS (MAIN)

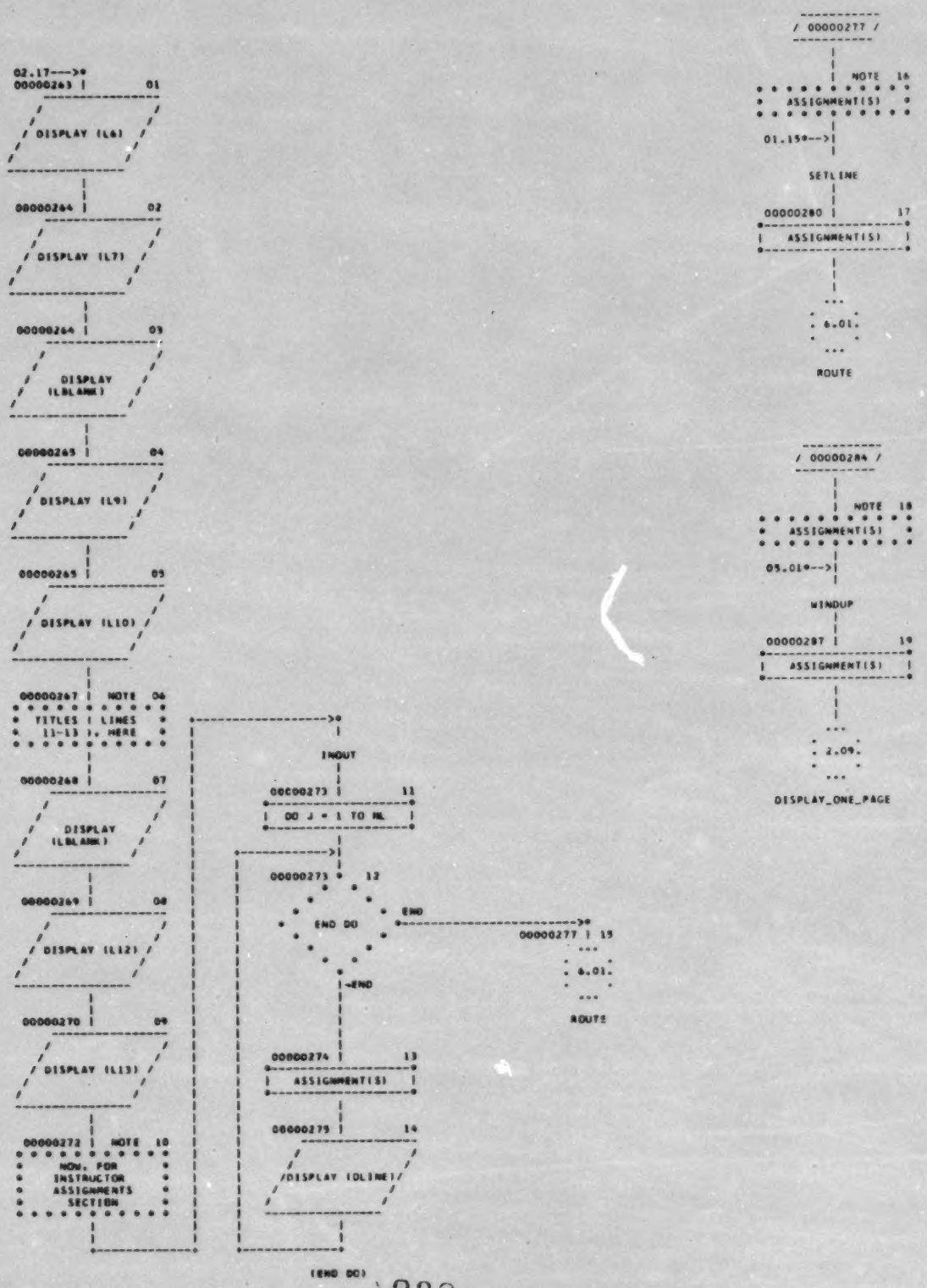


CHART TITLE - PROC PGM19 OPTIONS (MAIN)

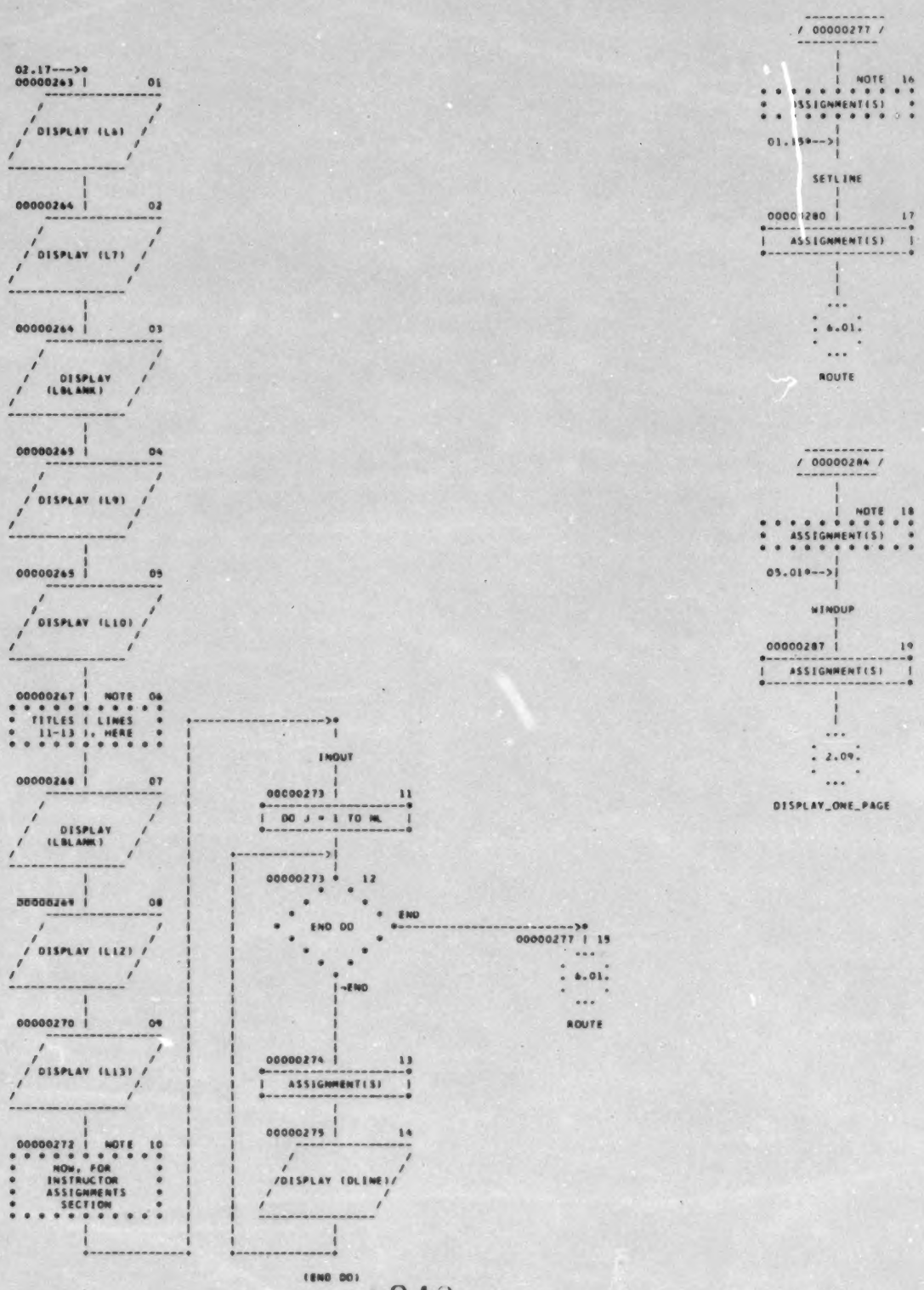


CHART TITLE - PROC PGM19 OPTIONS (MAIN)

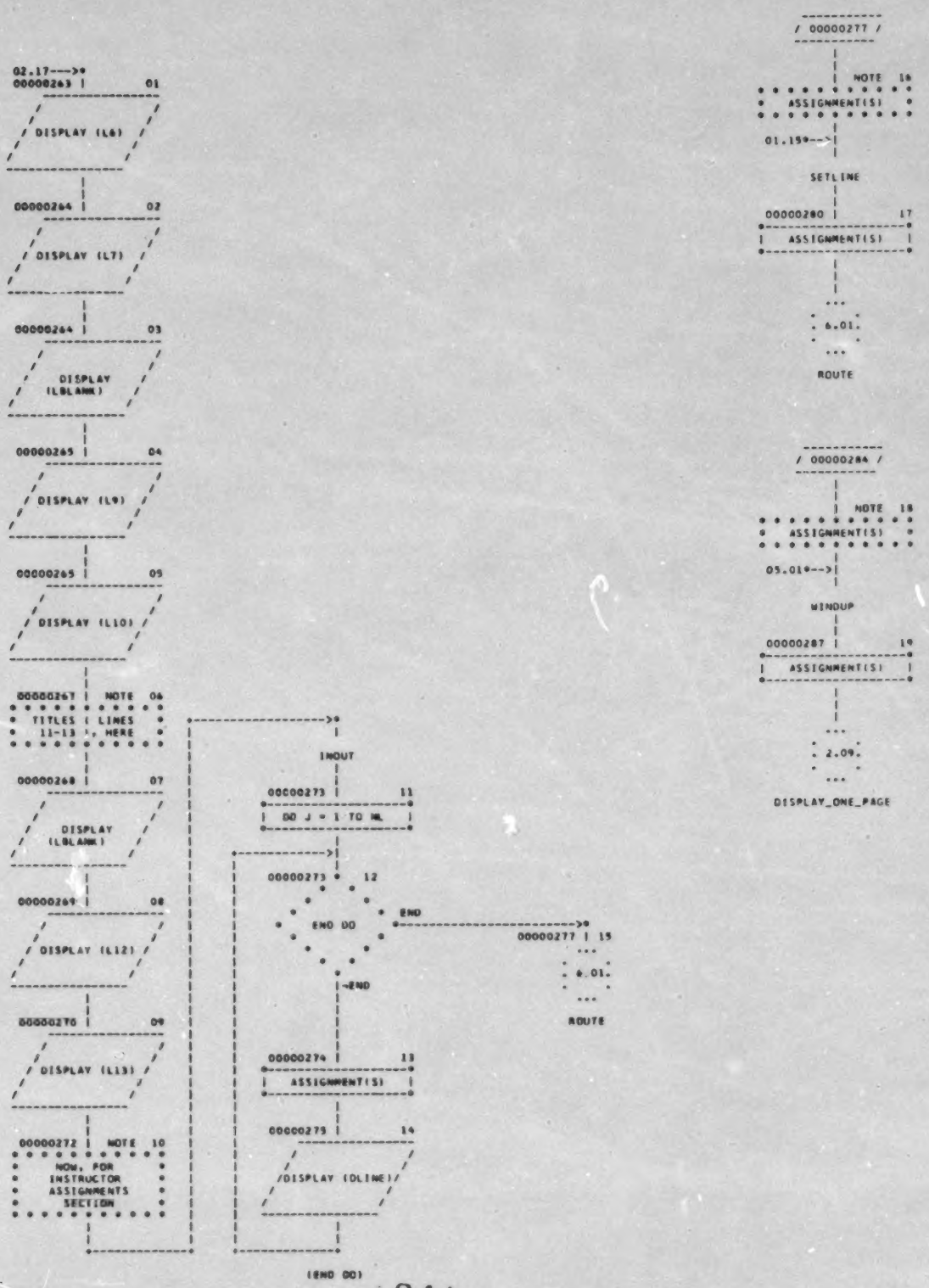


CHART TITLE - PROC PGM19 OPTIONS (MAIN)

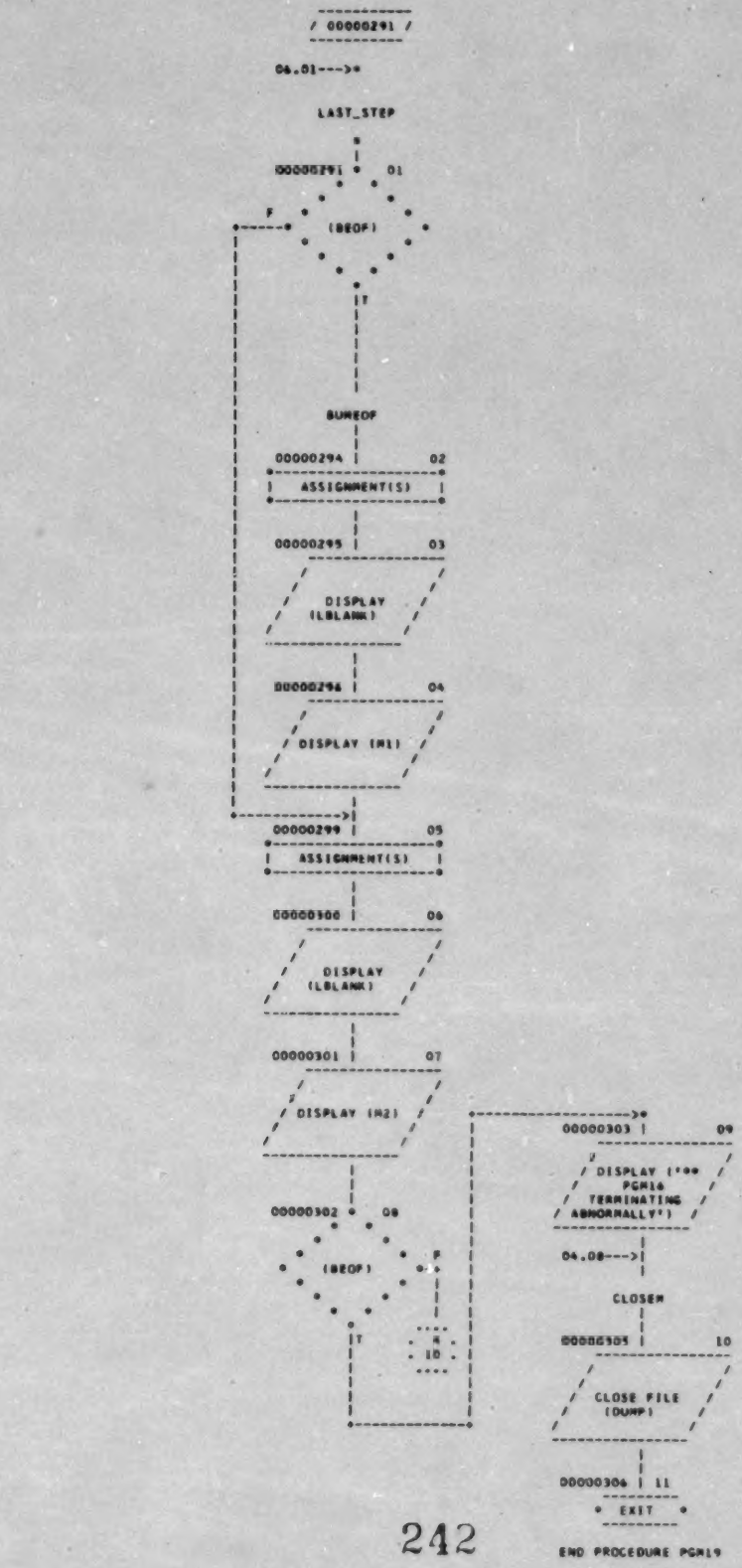


CHART TITLE - ON-UNIT ACTION BLOCKS

```
 ------------
/ 00000174 /
------------

01.03--->*

.ON-UNIT01

          *
00000174 |  NOTE  01
* * * * * * * * * * *
* ON ENDFILE (DUMP) *
* * * * * * * * * * *
          |
          |
          |
         ...
        .   .
        . 3.19.
        .   .
         ...

        WINDUP

   END .ON-UNIT01
```

CHART TITLE - LABEL ASSIGNMENT CROSS REFERENCE

```
        ----------
       /  ROUTE   /
       ----------
            |
 03.15*-->|
            |          01
*-------------------*
|      SCALAR       |
|  LABEL VARIABLE   |
*-------------------*
| SET AT  TO GO TO  |
|    1.11    2.02   |
| 00000186 START_NE |
|    1.15    2.01   |
| 00000193 SIR2     |
|    1.22    2.04   |
| 00000205 START_WH |
|    2.06    2.07   |
| 00000238 FILL_SI_ |
|    3.19    4.01   |
| 00000288 LAST_STE |
*-------------------*
```

PROGRAM NAME: PGM20

TYPE: PL/1

SYSTEM: Data Base Maintenance

NCSS FILENAME/FILETYPE: PGM20/PLI

A listing of the program source code and machine run flowcharts are included in this document on the pages following.

FUNCTION

Use of the RAMIS data maintenance function DUMP produces an 80 Byte (card image) output file, independent of the record length of the individual RAMIS records. As many card images per RAMIS record as required will be used to represent RAMIS records longer than 80 Bytes. For the Course data base, three (3) card-image records per RAMIS record are required.

For the Course data base, the field positions in the output of the DUMP are indicated in the following sample RAMIS output:

± EBCDIC RECORD LENGTH 8Ø CHARACTERS

FIELD NAME	FORMAT	LENGTH
SCH	A 4	4
CDP	A 4	4
DEL	A 1	1
CTLG	A 1Ø	1Ø
CRS	A 16	16
CONV1	I 3	3
LEN1	F4 .1	4
CAP1	I 3	3
BCAP1	I 3	3
DMD1	I 5	5
BDMD1	I 5	5
OFST	I 2	2
PFAIL	F5 .3	5
NADIS	F5 .3	5
NOSHOW	F5 .3	5
SETB	F5 .3	5
	X	Ø
BKLG	I 4	4
JNO	A 4	4
CONV2	I 3	3
LEN2	F4 .1	4
CAP2	I 3	3
BCAP2	I 3	3
DMD2	I 5	5
BDMD2	I 5	5
CHWK2	I 3	3
CONV3	I 3	3
LEN3	F4 .1	4
CAP3	I 3	3
BCAP3	I 3	3
DMD3	I 5	5

FIELD NAME	FORMAT	LENGTH
BDMD3	I 5	5
CHWK3	I 3	3
PAOB	I 4	4
AFAIL	F5 .3	5
PRTY	I 2	2
CTYPE	A 2	2
PNEC	A 4	4
UCAP	I 3	3
		X Ø
UDEM	I 3	3
FPRTY	I 2	2
RDTE	I 6	6
HSETB	F5 .3	5
TINSTR	F4 .1	4
SUIC	A 5	5
CCMM	A 1Ø	1Ø
SIR	F4 .1	4
CHRS	F5 .1	5
CTYP	A 1	1
ID	A 9	9
QUAL	A 3	3
ASN	I 1	1

Note that no slack bytes were required at the end of each of the first two card images at the positions marked 'X'. Only 58 Bytes were required on the third card image to fully represent the Course data base RAMIS record.

Inputs to RAMIS data management functions must be singular records. It is the function of PGM20 to combine the data from the three card images representing each RAMIS record into a single logical record.

For the Course file, using input as indicated above, the output record is composed of the following:

Logical Record Length: 218 Bytes

PL/1 Base Data Structure Name: RTRANS

PL/1 Output Filename: TRANS

First 80 Bytes of Output Record = Direct copy of first of three input card images

Second 80 Bytes of Output Record = Direct copy of the second of three input card images

Last 58 Bytes of Output Record = Direct copy of the first 58 columns of the last of the three input card images

USAGE

Program PGM 20 is invoked during operation of the following Root Executive Sequences (NCSS Filetype = EXEC):

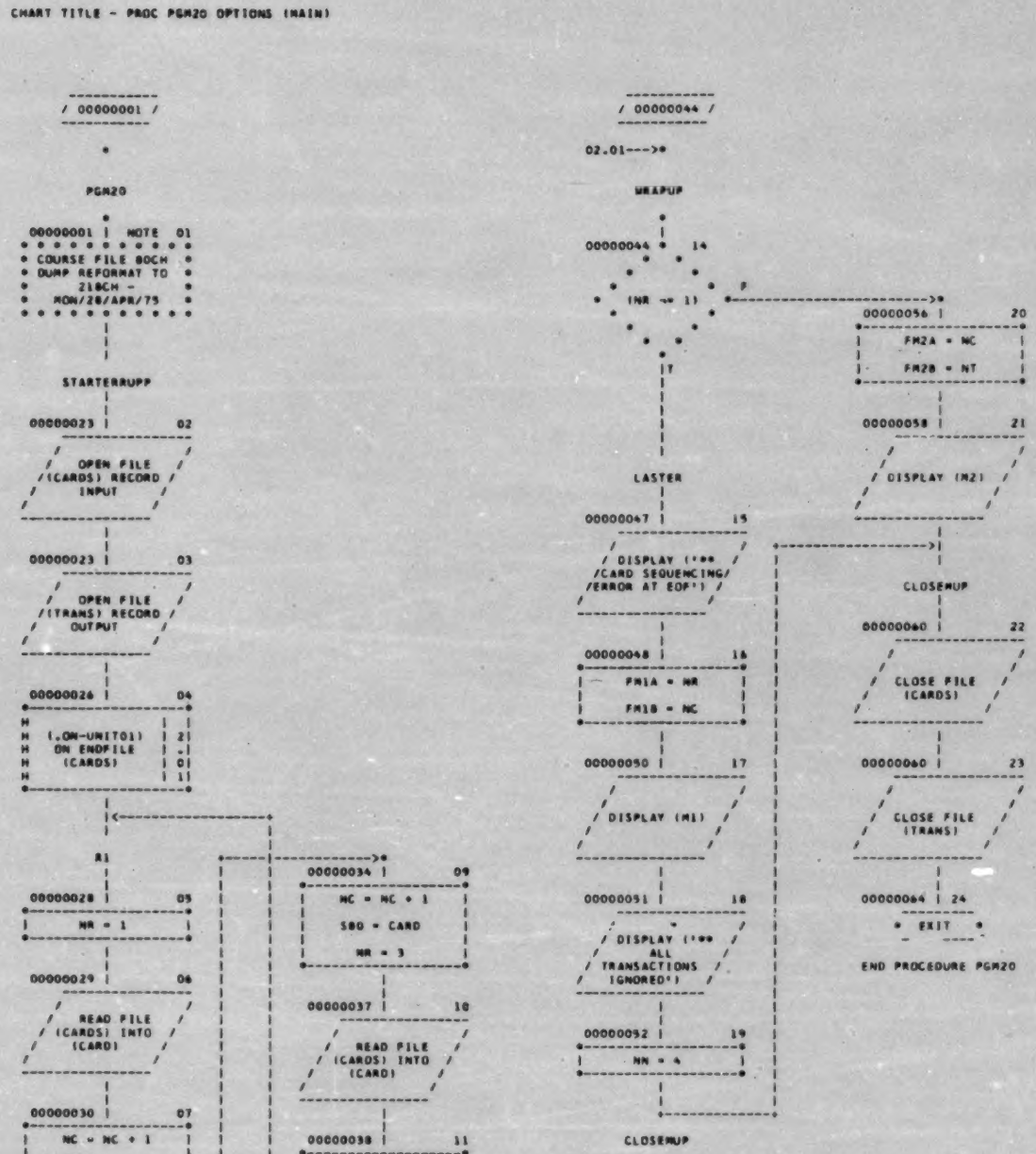
EXEC Sequence	Data Base
E50	Course
E51	Course

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PGM20- FLOWCHARTS

PL/I MODULE (LIST,WAMSG,SOURCSEQ)

```
CARD NO      ****                          CONTENTS                          ****

   1         PGM20:  /* COURSE FILE 80CH DUMP REFORMAT TO 218CH - MON/28/APR/75 */  00000100
   2                 PROC OPTIONS (MAIN) ;                                          00000200
   3                                                                                00000300
   4         DECLARE RTRANS     CHAR (218) STATIC INIT ( ' ' ), /* 218 USED */      00000400
   5                 FB0        CHAR (80) DEF RTRANS POS (1),                       00000500
   6                 SB0        CHAR (80) DEF RTRANS POS (81),                      00000600
   7                 TS8        CHAR (58) DEF RTRANS POS (161),                     00000700
   8                 CARD       CHAR (80) STATIC,                                   00000800
   9                 NC         FIXED BIN STATIC INIT ( 0 ),                        00000900
  10                 NR         FIXED BIN STATIC INIT ( 0 ),                        00001000
  11                 NT         FIXED BIN STATIC INIT ( 0 ),                        00001100
  12                 NN         FIXED BIN (31,0) STATIC INIT ( 0 ),                 00001200
  13                 IHESARC    ENTRY ( FIXED BINARY (31,0) ),                      00001300
  14                 M1         CHAR (52) STATIC INIT (                             00001400
  15         '** ERROR INDICATOR WAS -X- AT INPUT CARD COUNT X,XXX' ),              00001500
  16                 FM1A       PICTURE '9' DEF M1 POS (25),                        00001600
  17                 FM1B       PICTURE 'Z,ZZ9' DEF M1 POS (48),                    00001700
  18                 M2         CHAR (66) STATIC INIT (                             00001800
  19         '** TOTAL INPUT CARDS PROCESSED = X,XXX; TOTAL TRANSACTIONS = X,XXX' ),00001900
  20                 FM2A       PICTURE 'Z,ZZ9' DEF M2 POS (34),                    00002000
  21                 FM2B       PICTURE 'Z,ZZ9' DEF M2 POS (62) ;                   00002100
  22                                                                                00002200
  23         STARTERRUPP:                                                           00002300
  24                 OPEN FILE (CARDS) RECORD INPUT,                                00002400
  25                      FILE (TRANS) RECORD OUTPUT ;                              00002500
  26                 ON ENDFILE (CARDS) GO TO WRAPUP ;                              00002600
  27                                                                                00002700
  28         R1:     NR = 1 ;                                                       00002800
  29                 READ FILE (CARDS) INTO (CARD) ;                                00002900
  30                      NC = NC + 1 ;                                             00003000
  31                      FB0 = CARD ;                                              00003100
  32                 NR = 2 ;                                                       00003200
  33                 READ FILE (CARDS) INTO (CARD) ;                                00003300
  34                      NC = NC + 1 ;                                             00003400
  35                      SB0 = CARD ;                                              00003500
  36                 NR = 3 ;                                                       00003600
  37                 READ FILE (CARDS) INTO (CARD) ;                                00003700
  38                      NC = NC + 1 ;                                             00003800
  39                      TS8 = CARD ;                                              00003900
  40                           WRITE FILE (TRANS) FROM (RTRANS) ;                   00004000
  41                           NT = NT + 1 ;                                        00004100
  42                                GO TO R1 ;                                      00004200
  43                                                                                00004300
  44         WRAPUP:                                                                00004400
  45                 IF ( NR ¬= 1 ) THEN                                            00004500
  46         LASTER:     DO ;                                                       00004600
  47                      DISPLAY ( '** CARD SEQUENCING ERROR AT EOF' ) ;           00004700
  48                      FM1A = NR ;                                               00004800
  49                      FM1B = NC ;                                               00004900
  50                      DISPLAY ( M1 ) ;                                          00005000
  51                      DISPLAY ( '** ALL TRANSACTIONS IGNORED' ) ;               00005100
  52                           NN = 4 ;                                             00005200
  53                           GO TO CLOSEMUP ;                                     00005300
  54                      END LASTER ;                                              00005400
  55                                                                                00005500
  56                 FM2A = NC ;                                                    00005600
  57                 FM2B = NT ;                                                    00005700
  58                 DISPLAY ( M2 ) ;                                               00005800
  59                                                                                00005900
  60         CLOSEMUP:                                                              00006000
  61                 CLOSE FILE (CARDS),                                            00006100
  62                       FILE (TRANS) ;                                           00006200
  63                                                                                00006300
  64                 END PGM20 ;                                                    00006400
```

CHART TITLE - PROC PGM20 OPTIONS (MAIN)



A1 249

CHART TITLE - ON-UNIT ACTION BLOCKS

```
          -----------
         / 00000026 /
         -----------

 01.04--->*

.ON-UNIT01

                 *
 00000026 |  NOTE  01
* * * * * * * * * * *
*    ON ENDFILE     *
*      (CARDS)      *
* * * * * * * * * * *
          |
          |
          |
         ...
        .     .
        . 1.14.
        .     .
         ...

        WRAPUP

   END .ON-UNIT01
```

250

PROGRAM NAME: PGM22

TYPE: PL/1

SYSTEM: Data Base Maintenance

NCSS FILENAME/FILETYPE: PGM22/PLI

A listing of the program source code and machine run flowcharts are included in this document on the pages following.

FUNCTION

This program examines the weekly Instructor time cards and determines whether the time data can be successfully entered to the Instructor data base. This operation involves assuring that the Instructor name and ID have been entered into the data base. Also checked are the proper assignment of Courses to each Instructor.

USAGE

Program PGM22 is invoked during operation of the following Root Executive Sequences (NCSS Filetype = EXEC):

EXEC Sequence	Data Base
E22C	Instructor Master

INPUT FORMATS

The format of the weekly Instructor time card is shown on the following page in the illustration titled "Instructor Data Base Weekly Time Card Format". The time card file (filename - E22, filetype - SORT) will have been sorted via Executive Sequence E22A prior to run of program PGM22.

The second input file (filename = SPIN, filetype = DATA) to PGM22 is a reference file produced as output from Executive Sequence E22B. This file represents the current contents of the Instructor Master data base, and is structured as follows:

Logical Record Length: 17 Bytes

PL/1 Base Data Structure Name: RSPIN

Output PL/1 Filename: SPIN

Start Record Position	Byte Length	PL/1 Identifier	Format	Description
1	9	SID	A9	Instructor ID
10	4	SSCH	A4	Instructor School
14	4	SCDP	A4	Course CDP assigned to the indicated Instructor

One SPIN file record exists for each course assigned to each Instructor. Since the Instructor RAMIS data base is inherently sorted by Instructor ID and Course CDP, the SPIN file is sorted.

The Instructor time card file will be sorted to match the order of the SPIN file.

OUTPUT FORMATS

Three output files are produced via PGM22. They are:

NCSS Filename	NCSS Filetype	Logical Record	PL/1 Filename	Contents
E22	EMSG	61	EMSG	Error messages indicating discrepancies between time cards and data base contents
E22	BUST	80	ETRANS	Card images found to contain compare errors
E22	DATA	80	GOOD	Card images found to pass compare checks successfully

If no errors are detected in the processing of a batch of time cards, the error message and error card files (first two, above) will be empty. Conversely, if all cards are found to be in error, the E22/DATA file will be empty after PGM22 is run. Messages are output to the User terminal to indicate the exact results of PGM22 operation.

The two card-image files described above contain copies of the Instructor time cards; accordingly, the format of these two output files is the same as the input time card formats.

INSTRUCTOR DATA BASE WEEKLY TIME CARD FORMAT

INSTRUCTOR FILE

80-Column Card Punch Layout

IBM Federal Systems Division, Huntsville Systems Center, Huntsville, Alabama

NAME: INSTRUCTOR TIME CARD

ID	SCH	CDP	A1	A2	A3	A4	A5	A6	A7	B1	B2	B3	B4	B5	B6	B6A

INSTRUCTOR TIME(S), AS REQUIRED = XX_AX

A = LOCATION OF ASSUMED DECIMAL POINT

955-1115-0

PL/I MODULE (LIST,NAMSQ,SOURCSEQ)

```
CARD NO    ****                          CONTENTS                          ****

   1       OPTION PACKIF,COMPRESS
   2       PGM22:  /* TIME CARD PRE-PROCESSOR - MON/30/JUNE/75 */
   3               PROC OPTIONS (MAIN) ;
   4
   5       DECLARE CARD      CHAR (80) STATIC INIT ( ' ' ),
   6               CID       CHAR (9) DEF CARD POS (1),
   7               CSCH      CHAR (4) DEF CARD POS (10),
   8               CCDP      CHAR (4) DEF CARD POS (14),
   9               CF17      CHAR (17) DEF CARD POS (1),
  10               RSPIN     CHAR (17) STATIC INIT ( ' ' ),
  11               SID       CHAR (9) DEF RSPIN POS (1),
  12               SSCH      CHAR (4) DEF RSPIN POS (10),
  13               SCDP      CHAR (4) DEF RSPIN POS (14) ;
  14
  15       DECLARE (NC, NS, NER, NH, NOID, NOSCH, NOCDP
  16                        ) FIXED BIN STATIC,
  17               MLINE     CHAR (61) STATIC INIT ( 'NOMIT ' ),
  18               ET        CHAR (3) DEF MLINE POS (7),
  19               EC        CHAR (17) DEF MLINE POS (19),
  20               ES        CHAR (17) DEF MLINE POS (45),
  21               CCN       PICTURE 'Z,ZZ9' DEF MLINE POS (12),
  22               SSN       PICTURE 'Z,ZZ9' DEF MLINE POS (38),
  23               ROUTE     LABEL ;
  24
  25       DECLARE FLINE1    CHAR (77) STATIC INIT ( ' ' ),
  26               FLINE2    CHAR (61) STATIC INIT ( ' ' ),
  27               PNC       PICTURE 'Z,ZZ9' DEF FLINE1 POS (18),
  28               PNH       PICTURE 'Z,ZZ9' DEF FLINE1 POS (41),
  29               PNOID     PICTURE 'Z,ZZ9' DEF FLINE1 POS (57),
  30               PNOCDP    PICTURE 'Z,ZZ9' DEF FLINE1 POS (73),
  31               PNS       PICTURE 'Z,ZZ9' DEF FLINE2 POS (18),
  32               PNER      PICTURE 'Z,ZZ9' DEF FLINE2 POS (41),
  33               PNOSCH    PICTURE 'Z,ZZ9' DEF FLINE2 POS (57) ;
  34
  35       STARTERRUPP:
  36               OPEN FILE (CARDS) RECORD INPUT,
  37                    FILE (SPIN) RECORD INPUT,
  38                    FILE (GOOD) RECORD OUTPUT,
  39                    FILE (EMSG) RECORD OUTPUT,
  40                    FILE (ETRANS) RECORD OUTPUT ;
  41               ON ENDFILE (CARDS) GO TO WINDUP ;
  42               ON ENDFILE (SPIN) GO TO DONETOOSOON ;
  43               ROUTE = R1 ;
  44               NC, NS, NER, NH, NOID, NOSCH, NOCDP = 0 ;
  45               FLINE1 = 'CARDS (NC)      : X,XXX   HITS (NH)'
  46                    || '    : X,XXX   NOID  : X,XXX   NOCDP '
  47                    || ': X,XXX' ;
  48               FLINE2 = 'SPIN RECS (NS) : X,XXX   ERRORS (NER) '
  49                    || ': X,XXX   NOSCH : X,XXX' ;
  50
  51               READ FILE (SPIN) INTO (RSPIN) ;
  52                    NS = 1 ;
  53
  54       R1:     READ FILE (CARDS) INTO (CARD) ;
  55                    NC = NC + 1 ;
  56       R1A:    IF ( CID = SID ) THEN GO TO STEP2 ;
  57               IF ( CID < SID ) THEN GO TO NOMIT_ID ;
  58                    READ FILE (SPIN) INTO (RSPIN) ;
  59                    NS = NS + 1 ;
  60                         GO TO R1A ;
  61
  62       STEP2:  /* HERE, ID NUMBERS ARE EQUAL */
  63                    IF ( CCDP = SCDP ) THEN GO TO STEP3 ;
  64          /* HERE, ID NUMBER HIT, CDP HIT, NOW TEST SCHOOL */
  65                    IF ( CSCH ¬= SSCH ) THEN GO TO NOMIT_SCH ;
  66          /* HERE, HITS ALL THE WAY... */
  67                    NH = NH + 1 ;
  68                    WRITE FILE (GOOD) FROM (CARD) ;
  69                         GO TO R1 ;
  70
  71       STEP3:  /* HERE ID NUMBER HIT, CDP MISMATCH - TRY ANOTHER CDP */
  72               READ FILE (SPIN) INTO (RSPIN) ;
  73                    NS = NS + 1 ;
  74               IF ( CID ¬= SID ) THEN GO TO NOMIT_CDP ;
  75                    GO TO STEP2 ;
  76
  77       NOMIT_ID:     NOID = NOID + 1 ;  ET = 'ID' ;  GO TO BUSTED ;
  78       NOMIT_CDP:    NOCDP = NOCDP + 1 ;  ET = 'CDP' ;  GO TO BUSTED ;
  79       NOMIT_SCH:    NOSCH = NOSCH + 1 ;  ET = 'SCH' ;
  80
  81       BUSTED:
  82               NER = NER + 1 ;  EC = CF17 ;  ES = RSPIN ;
  83               CCN = NC ;       SSN = NS ;
  84                    WRITE FILE (EMSG) FROM (MLINE) ;
  85                    WRITE FILE (ETRANS) FROM (CARD) ;
  86                         GO TO ROUTE ;
  87
  88       DONETOOSOON:
  89               ROUTE = STEP4 ;
  90               ET = 'EOF' ;
  91               ON ENDFILE (CARDS) GO TO WINDUP ;
  92                    GO TO BUSTED ;
  93
  94       STEP4:  READ FILE (CARDS) INTO (CARD) ;
  95               NC = NC + 1 ;
  96                    GO TO BUSTED ;
  97
  98       WINDUP:
  99               CLOSE FILE (CARDS), FILE (SPIN), FILE (GOOD),
 100                     FILE (EMSG), FILE (ETRANS) ;
 101
 102               DISPLAY ( 'TIME CARD PRE-PROCESSING COMPLETE' ) ;
 103               PNC = NC ;  PNH = NH ;  PNOID = NOID ;  PNOCDP = NOCDP ;
 104               PNS = NS ;  PNER = NER ;  PNOSCH = NOSCH ;
 105               DISPLAY ( FLINE1 ) ;
 106               DISPLAY ( FLINE2 ) ;
 107
 108                    END PGM22 ;
```

CHART TITLE - PROC PGM22 OPTIONS (MAIN)

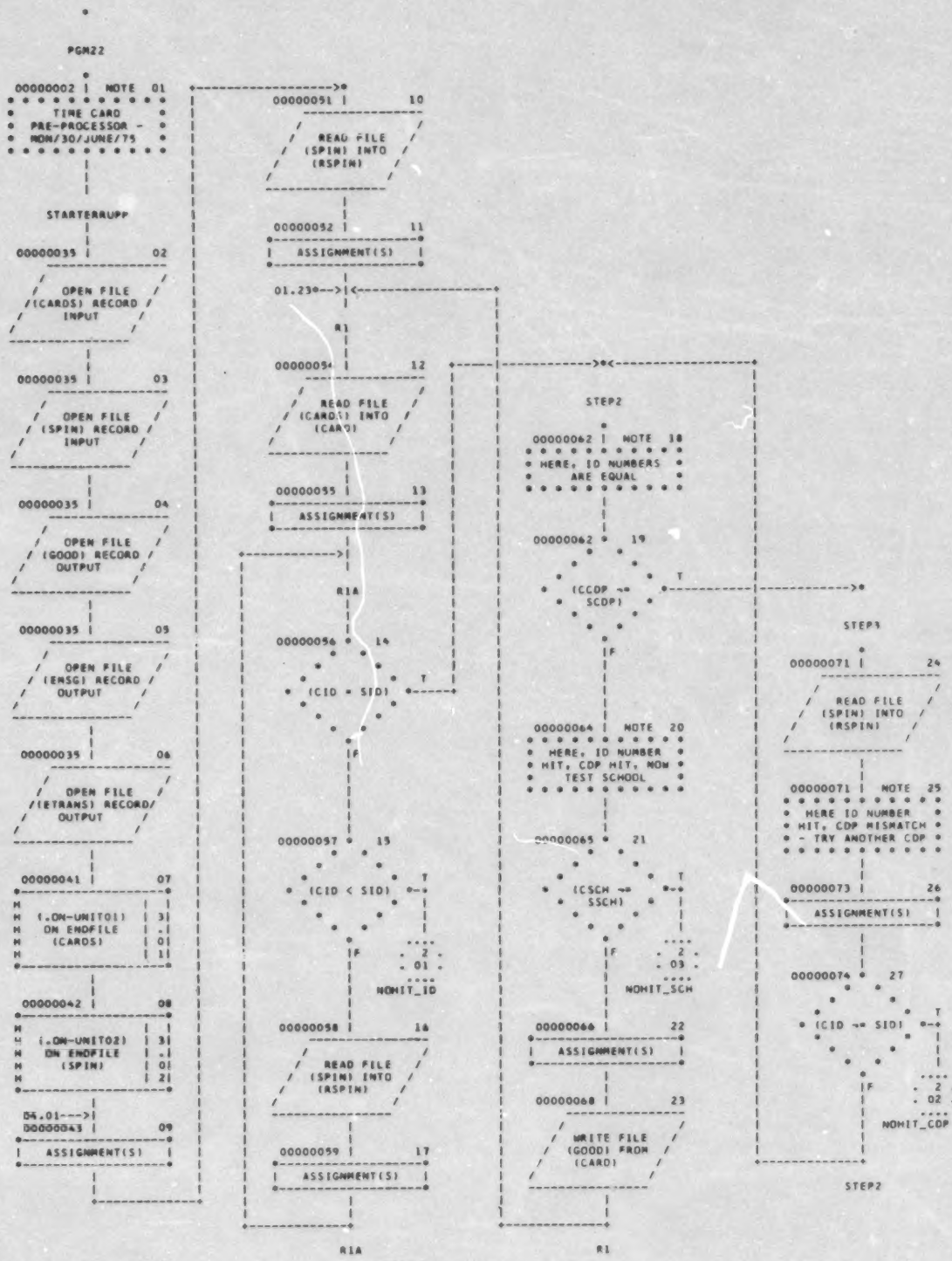


CHART TITLE - PROC PGM22 OPTIONS (MAIN)

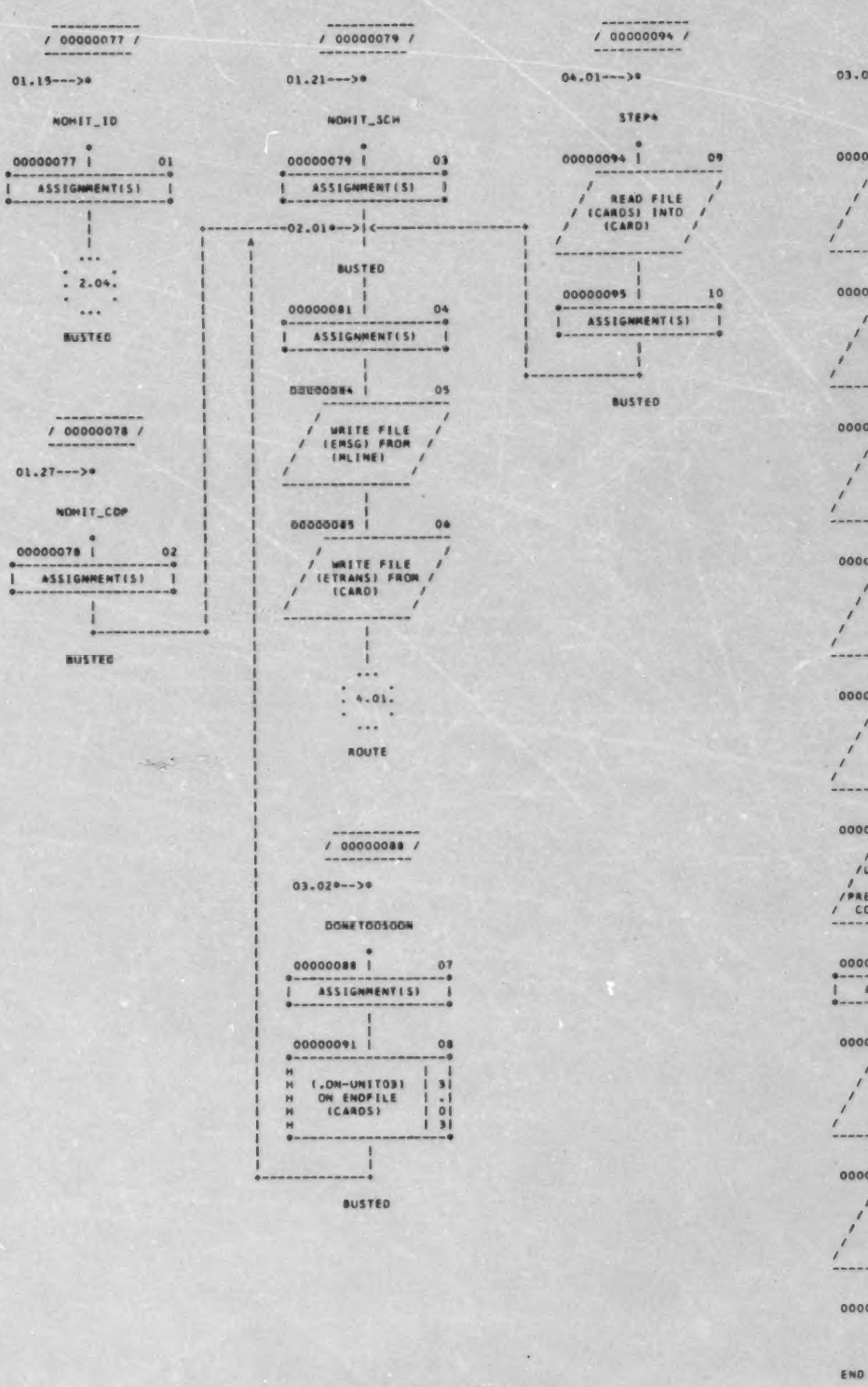


CHART TITLE - ON-UNIT ACTION BLOCKS

```
          -----------
         / 00000041 /
         -----------

    01.07--->*

   .ON-UNIT01

                 *
    00000041 |  NOTE  01
   * * * * * * * * * * *
   *    ON ENDFILE     *
   *      (CARDS)      *
   * * * * * * * * * * *
              |
              |
              |
             ...
            .   .
            . 2.11.
            .   .
             ...

           WINDUP

      END .ON-UNIT01


          -----------
         / 00000042 /
         -----------

    01.08--->*

   .ON-UNIT02

                 *
    00000042 |  NOTE  02
   * * * * * * * * * * *
   * ON ENDFILE (SPIN) *
   * * * * * * * * * * *
              |
              |
              |
             ...
            .   .
            . 2.07.
            .   .
             ...

         DONETOOSOON

      END .ON-UNIT02


          -----------
         / 00000091 /
         -----------

    02.08--->*

   .ON-UNIT03

                 *
    00000091 |  NOTE  03
   * * * * * * * * * * *
   *    ON ENDFILE     *
   *      (CARDS)      *
   * * * * * * * * * * *
              |
              |
              |
             ...
            .   .
            . 2.11.
            .   .
             ...

           WINDUP

      END .ON-UNIT03
```

CHART TITLE - LABEL ASSIGNMENT CROSS REFERENCE

```
        ----------
       /  ROUTE   /
       ----------
            |
 02.06--->|
            |          01
*-------------------*
|      SCALAR       |
|  LABEL VARIABLE   |
*-------------------*
| SET AT  TO GO TO  |
|    1.09    1.12   |
| 00000043 R1       |
|    2.07    2.09   |
| 00000089 STEP4    |
*-------------------*
```

PROGRAM NAME: PROMPTX

TYPE: PL/1

SYSTEM: Data Base Maintenance

NCSS FILENAME/FILETYPE: PROMPT/PLI

A listing of the program source code and machine run flowcharts are included in this document on the pages following.

FUNCTION

This program is used in conjunction with each of the Executive Sequences termed "Class A Prompting Executives". Such Executive Sequences are used for data base maintenance (entry, change, deletion).

The PROMPTX program will direct the User at the terminal as to which data fields to enter. As a result of the manner in which the prompting text has been composed, the field format will also be a portion of the prompt. Some fields are mandatory, some optional. In the case of the optional types, the User may elect to let the field value default rather than provide a specific data entry from the terminal. A keyboard entry has been provided to allow the User to abort the data maintenance operation; such a request effectively cancels any further data maintenance operation that may have resulted.

The actions to be taken by PROMPTX in each of the Executive Sequences mentioned below is directed by a unique table input for each Sequence. The format of this table is discussed under INPUT FORMAT, below.

USAGE

Program PROMPTX is invoked during operation of the following Root Executive Sequences (NCSS Filetype = EXEC):

EXEC Sequence	Data Base
E1, W1	Facilities
E2,W2	
E20, W20	Instructor
E21, W21	
E53, W53	
E54, W54	
E40, W40	Course
E41, W41	
E42, W42	
E64, W64	

INPUT FORMAT

PROMPTX is a table-driven program. The input table provides the text to be used in prompting the User at the terminal. Also included in the input table is data which directs PROMPTX in the validation of terminal input. The structure of the input file is:

Logical Record Length: 80 Bytes

PL/1 Base Data Structure Name: CARD

Input PL/1 Filename: TAB

Card Columns	Format	Contents/Meaning
*** TYPE -0- CARD ***		
1	A1	'0' means Type -0- Card
2/9	A8	'ACTION' Signifies "Action" Card
10	A1	'1' = PROMPTX program will be allowed to function in both NEW and CHANGE operational modes
		'2' = PROMPTX allowed to function only in NEW mode
11/12	A2	'1' = Special processing required at start of PROMPTX. Any entry other than '1', implies no special processing on entry.
13/14	A2	'1' = Special processing required on exit from PROMPTX (before writing the completed data maintenance transaction). Entry other than '1' indicates no such processing on exit.

Columns	Format	Contents/Meaning
*** TYPE -1- CARD ***		
1	A1	'1' = Signifies Type -1- Card format
2/41	A40	Text to be output to the User terminal when PROMPTX commences operation. This text is output only once - prior to prompting for any data fields. As many lines of entry text may be included as desired. Note maximum 40 characters width.
		NOTE - PROMPTX has internal array space for 100 lines of text (40 characters per line) - this must suffice for all entry text (Type -1- cards) plus all text required for data field prompting (Type -3- cards).
*** TYPE -3- CARD ***		
1	A1	'3' = Signifies Type -3- Card format
2/41	A40	Text to be output when a data field is to be prompted. Line width is 40 characters max. Number of lines may vary - see restrictions on the total number of text lines under the description for the Type -1- card format.
*** TYPE -2- CARD ***		
1	A1	'2' = Signifies Type -2- Card format
2/11	A10	Mnemonic name associated with the data field described on this card
12/13	Unused	
14	I1	Field Type: 1 = Character 2 = Integer 3 = Fraction
15/16	I2	For character fields only; maximum number of characters allowed. Zero (0) means don't care.
17/18	Unused	
19/34	A16	Default value for data field, if used; otherwise blank.

Columns	Format	Contents/Meaning
*** TYPE -2-, Continued ***		
35	I1	Mandatory/optional flag; 1 = Cannot omit this field; User must supply a value from the terminal after prompting 2 = Can omit this field if User so indicates; use standard default (numeric = 0, alphabetic = blank). 3 = Can omit; use default provided in card columns 19/34
36	I1	Limit check flag for numeric fields: 1 = No limit check 2 = Check both upper and lower limits 3 = Check only high limit 4 = Check only lower limit
34/46	Numeric, may be signed and/or fractional = Lo limit numeric check, if used	
47/56	Numeric, may be signed and/or fractional = Hi limit check for numeric field, if used	
Note - Fields in card positions 57 - 63, inclusive, indicate the position and format of the data field in the output transaction. PROMPTX will place the default or verified User input data into this location. . . .		
57/58	I2	PL/1 PUT STRING verb remote format statement selector. Numeric fields will be converted to this format before entry to the output transaction: 01 = PL/1 Format F(6,1) 02 = PL/1 Format F(5,1) 03 = PL/1 Format F(3,1) 04 = PL/1 Format F(4,1) 05 = PL/1 Format F(5,3)
59/61	I3	Starting field position in the output transaction; legal values are 1 - 250
62/63	I2	Length of the data field in the output transaction; length plus starting position minus one cannot exceed 250

INPUT TABLE LAYOUT

Following are the rules to be followed in structuring a table for PROMPTX. The card formats were discussed above; they will be referenced below only by card type (0, 1, 2, or 3).

All input table card images must appear in the following order:

(1) One (only) Type -0- card. If used, this must be first. Type -0- cards are valid only for the data fields associated with the Instructor data base.

(2) As many Type -1- entry text cards as desired. If desired, Type -1- cards may be omitted altogether.

From one to 55 data fields may be prompted; the order of appearance in the input table will be the prompt order. Each data field prompt will be signified by:

One Type -2- card describing the data field.

One or more Type -3- cards representing one line per card of prompting text to be presented to the User.

The Type -2- card must precede the Type -3- card(s) for each data field. At least one of each must represent each data field.

Current maximum limitations for the PROMPTX program are:

55 data fields maximum

100 text lines distributed between the entry text and the prompting text for the data fields.

OUTPUT FORMAT FOR PROMPTX

Logical Record Length: 250 Bytes

PL/1 Base Data Structure Name: RTRANS

PL/1 Filename: TRANS (Attributes are RECORD UPDATE SEQUENTIAL)

Output file transaction format is as described in the reference table input.

If NEW is selected by the User as the PROMPTX operational mode, the transaction is blanked before filling with new data.

If CHANGE is selected as the operational mode, the single record of the transaction file is read into store before proceeding. This allows the User to change any/all fields or merely examine the fields' contents.

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PROMPTX- FLOWCHARTS

PL/1 MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
 1   OPTION PACKIF,COMPRESS
 2   PROMPTX:/* GP FIELD ENTRY AND TRANSACTION PROCESSOR - WED/16/APR/75 */
 3          PROC OPTIONS (MAIN) ;
 4
 5   DECLARE ( F0(55), FT(55), MAXC(55), DSS(55), OMIT(55), LIMIT(55),
 6             FSEL(55), START(55), LEN(55), NLINE(55), LISS(55),
 7             NT, NER, NFLDS, M, NN, X, MMAX, J, L, MODE )
 8                FIXED BIN (15) STATIC ;
 9
10   DECLARE NN (55)   CHAR (12) STATIC INIT ( (55)(12)' ' ),
11           DEF (55)  CHAR (16) STATIC INIT ( (55)(16)' ' ),
12           ( LLO(55), LHI(55) ) FIXED DEC (8,2) STATIC,
13           NEW       BIT (1) STATIC,
14           PTEXT     FIXED BIN (15) STATIC INIT ( 1 ),
15           NETXT     FIXED BIN (15) STATIC INIT ( 0 ),
16           N         FIXED BIN (15) STATIC INIT ( 0 ),
17           ( SAMP, LSAMP ) FIXED DEC (12,2) STATIC ;
18
19   DECLARE TEXT (100) CHAR (40) STATIC INIT ( (100)(40)' ' ),
20           CARD      CHAR (80) STATIC,
21           CTYPE     CHAR (1) DEF CARD POS (1),
22           T40       CHAR (40) DEF CARD POS (2),
23           F40       CHAR (40) DEF CARD POS (1),
24           F70       CHAR (70) DEF CARD POS (1),
25           CTACT     CHAR (7) DEF CARD POS (1),
26           LBLANK    CHAR (10) STATIC INIT ( '*' ),
27           RTRANS    CHAR (250) STATIC,
28           TIMETYP   CHAR (3) DEF RTRANS POS (80),
29           T100      CHAR (100) DEF RTRANS POS (1),
30           T200      CHAR (100) DEF RTRANS POS (101),
31           T50       CHAR (50) DEF RTRANS POS (201) ;
32
33   DECLARE REP       CHAR (80) STATIC,
34           REP1      CHAR (1) DEF REP POS (1),
35           TRY12     CHAR (2) DEF REP POS (1),
36           REP2      CHAR (2) DEF REP POS (1),
37           TRYOUT    CHAR (3) DEF REP POS (1),
38           TRYS      CHAR (4) DEF REP POS (1),
39           TRYC      CHAR (6) DEF REP POS (1),
40           REPTAB (80) CHAR (1) BASED (PR),
41           TRY16     CHAR (16) STATIC,
42           ( ROUTE, WAY, FF ) LABEL,
43           FORMS (10) LABEL
44                 INIT ( RF1, RF2, RF3, RF4, RF5 ) ;
45
46   DECLARE TRYM      CHAR (12) STATIC,
47           DERM      CHAR (27) STATIC INIT
48                 ( '*** ERROR ONCODE WAS XXXX *' ),
49           NONC       PICTURE '9999' DEF DERM POS (22),
50           ERMSG     CHAR (26) STATIC INIT
51                 ( '*** ERROR # NN OCCURED ***' ),
52           NNER       PICTURE 'Z9' DEF ERMSG POS (13),
53           FNME      CHAR (12) STATIC,
54           FN12 (12) CHAR (1) BASED (PF),
55           VMSG      CHAR (39) STATIC INIT
56                 ( '*    CURRENT VALUE IS =' ),
57           FILLV     CHAR (1) DEF VMSG POS (24) ;
58
59   DECLARE LIMSG     CHAR (47) STATIC INIT
60                 ( '* VALUE SUPPLIED OUTSIDE XX-LIMIT OF XXXXXXXX *' ),
61          TYPLIM    CHAR (2) DEF LIMSG POS (26),
62           ILIMVAL   PICTURE 'ZZZZZZZZ9' DEF LIMSG POS (38),
63           FLIMVAL   PICTURE 'ZZZZ9V.99' DEF LIMSG POS (38) ;
64
65   DECLARE IDEF      CHAR (84) STATIC INIT (
66   '175000001 A2 A3 A4 A5 A6 A7 B1 B2 B3 B4 B5 B6 B6A00.0000.00000.0' ),
67           TIDEF     CHAR (84) DEF RTRANS POS (73),
68           FIRST     BIT (1) STATIC INIT ( '1'B ),
69           IMESARC   ENTRY (FIXED BINARY(31,0)),
70           NORM      FIXED BIN (15) STATIC INIT (1),
71           BSTAND    FIXED BIN (15) STATIC INIT (0),
72           ASTAND    FIXED BIN (15) STATIC INIT (0) ;
73
74   /* FOLLOWING ARE THE REMOTELY-CALLED FORMAT STATEMENTS */
75   RF1:     FORMAT    ( F(6,1) ) ;
76   RF2:     FORMAT ( F(5,1) ) ;
77   RF3:     FORMAT ( F(3,1) ) ;
78   RF4:     FORMAT ( F(4,1) ) ;
79   RF5:     FORMAT ( F(5,3) ) ;
80
81   /*** ERROR ONCODE TRAP HERE ***/
82
83           ON ERROR BEGIN ;
84                    NONC = ONCODE ;
85                    DISPLAY ( DERM ) ;
86                    IF ( FIRST ) THEN DISPLAY ( F70 ) ;
87                         NN = 8 ;
88                         GO TO SETCODE ;
89                    END ;
90
91   /*** DESCRIPTION FILE IS LOADED HERE ***/
92
93           OPEN FILE (TAB) RECORD INPUT ;
94           ON ENDFILE (TAB) GO TO STARTERUP ;
95           NN = 0 ;
96
97   ER7:    NER = 7 ;
98           READ FILE (TAB) INTO (CARD) ;
99           IF ( CTACT ¬= 'OACTION' ) THEN GO TO R1A ;
100                GET STRING (CARD) EDIT ( NORM, BSTAND, ASTAND )
101                     ( X(9), F(1), F(2), F(2) ) ;
102
103  ER1:    NER = 1 ;
104  R1:     READ FILE (TAB) INTO (CARD) ;
105  R1A:    IF ( CTYPE = '1' ) THEN
106  L1:          DO ;
107                NETXT = NETXT + 1 ;
108                NN = NN + 1 ;
109                IF ( NN > 100 ) THEN GO TO [illegible]
```

```
                 TEXT (NN) = T40 ;
                       GO TO R1 ;
                 END L1 ;
ER2:    NER = 2 ;
ER2A:   IF ( CTYPE ¬= '2' ) THEN GO TO BUST ;
ER6:    NER = 6 ;
        N = N + 1 ;
        IF ( N > 55 ) THEN GO TO BUST ;

        GET STRING (CARD) EDIT
              ( MN(N), F0(N), FT(N), MAXC(N), DSS(N), DEF(N),
                OMIT(N), LIMIT(N), LLO(N), LHI(N), FSEL(N),
                START(N), LEN(N), NLINE(N), LISS(N) )
        ( X(1),A(10), F(2), F(1), F(2), F(2), A(16),
              F(1), F(1), F(10,2), F(10,2),
              F(2), F(3), F(2), F(1), F(2) ) ;

        READ FILE (TAB) INTO (CARD) ;
ER3:    NER = 3 ;
        IF ( CTYPE ¬= '3' ) THEN GO TO BUST ;

        NN = NN + 1 ;
ER4:    NER = 4 ;
        IF ( NN > 100 ) THEN GO TO BUST ;
        K = 1 ;
        LISS (N) = NN ;
              GO TO L2A ;

L2:     IF ( CTYPE ¬= '3' ) THEN GO TO L2B ;
              NN = NN + 1 ;
              IF ( NN > 100 ) THEN GO TO BUST ;
              K = K + 1 ;
L2A:          TEXT (NN) = T40 ;
                    NLINE (N) = K ;
              READ FILE (TAB) INTO (CARD) ;
                    GO TO L2 ;

L2B:
ER5:    NER = 5 ;
              GO TO ER2A ;

/*** HERE, DESCRIPTION TABLE EOF: INITIAL ROUTINE ENTRY PROMPTED ***/

STARTERUP:
        NMAX = N ;
        CLOSE FILE (TAB) ;
        FIRST = '0'B ;

            ( NETXT < 1 ) THEN GO TO WHICHWAY ;
L3:           DO J = 1 TO NETXT ;
              DISPLAY ( TEXT(J) ) ;
              END L3 ;

WHICHWAY:
        OPEN FILE (TRANS) SEQUENTIAL UPDATE ;
        READ FILE (TRANS) INTO (RTRANS) ;

        ON CONVERSION BEGIN ;
                    NONC = ONCODE ;
                    DISPLAY ( DERM ) ;
                          GO TO NEWLOOP ;
                    END ;

STEP1:
        DISPLAY ( LBLANK ) ;
        IF ( NORM ¬= 1 ) THEN GO TO NEWSTUFF ;
        DISPLAY ( LBLANK ) ;
        DISPLAY ( '* SPECIFY EITHER NEW OR CHANGE *' )
              REPLY ( REP ) ;
        IF ( TRYOUT = 'NEW' ) THEN GO TO NEWSTUFF ;
        IF ( TRYC = 'CHANGE' ) THEN GO TO CHANGEIT ;
        IF ( TRYOUT = 'OUT' ) THEN GO TO OUTNOW ;

        ROUTE = STEP1 ;
VOSIS:  DISPLAY ( '* CANNOT RECOGNIZE LAST ENTRY - TRY AGAIN *' ) ;
              GO TO ROUTE ;

/*** BRAND NEW TRANSACTIONS MADE UP IN THIS SECTION ***/

NEWSTUFF:
        NEW = '1'B ;
        N = 1 ;
        RTRANS = ' ' ;
        IF ( BSTAND = 1 ) THEN
              TIDEF = IDEF ;

NEWLOOP:
      IF ( ¬NEW ) THEN GO TO OLDE ;
        L = NLINE (N) ;
L4A:    J = LISS (N) - 1 ;
        DISPLAY ( LBLANK ) ;
L4:           DO K = 1 TO L ;
              J = J + 1 ;
              IF ( K = L ) THEN GO TO STEP2 ;
              DISPLAY ( TEXT (J) ) ;
              END L4 ;

STEP2:  DISPLAY ( TEXT (J) )
              REPLY (REP) ;
        IF ( TRYOUT = 'OUT' ) THEN GO TO OUTNOW ;

        NN = N ;
        IF ( TRYS = 'DONE' ) THEN
MAYBDONE:     DO N = NN TO NMAX ;
              IF ( OMIT(N) ¬= 1 ) THEN GO TO MAY1 ;
              DISPLAY ( '* CANNOT OMIT FIELD CALLED ' || MN(N) ) ;
                    GO TO NEWLOOP ;
MAY1:         SUBSTR ( RTRANS, START(N), LEN(N) ) = DEF (N) ;
              IF ( N = NMAX ) THEN GO TO FINISH_TRANSACTION ;
              END MAYBDONE ;
```

```
CARD NO      ****                    CONTENTS                    ****

 222                IF ( TRYS = 'N') | (TRYS = 'NO' ) | (TRYS = 'NONE') THEN
 223        L5:         DO ;
 224                     IF ( OMIT(N) ¬= 1 ) THEN GO TO STEP3 ;
 225                     DISPLAY ( '* CANNOT OMIT THIS ENTRY -TRY AGAIN *' ) ;
 226                         GO TO NEWLOOP ;
 227        STEP3:      SUBSTR ( RTRANS, START(N), LEN(N) ) = DEF (N) ;
 228                         GO TO TALLY ;
 229                     END L5 ;
 230
 231        LOOKAT: TRY16 = REP ;
 232        CHARACTER_FIELD:
 233                IF ( FT(N) = 1 ) THEN
 234        L6:         DO ;
 235                     J = MAXC (N) + 1 ;
 236                     IF ( J >= 15 ) THEN GO TO MAKETRANS ;
 237                     K = INDEX ( TRY16, '  ' ) ;
 238                IF ( ( K > 0 ) & ( K <= J ) ) THEN GO TO MAKETRANS ;
 239                     DISPLAY ( '* MAX NUMBER CHARACTERS EXCEEDED *' ) ;
 240                     GO TO NEWLOOP ;
 241                     END L6 ;
 242
 243        NUMERIC_FIELD:
 244                L = LEN (N) ;
 245                J = FT (N) ;
 246                K = LIMIT (N) ;
 247
 248                NT = INDEX ( TRY16, '  ' ) - 1 ;
 249                IF ( NT = 0 ) THEN SIGNAL CONVERSION ;
 250                IF ( INDEX(TRY16,'-') ¬= 0 ) THEN SIGNAL CONVERSION ;
 251                IF ( INDEX ( SUBSTR(TRY16,1,NT), '.' ) = 0 )
 252                     THEN GET STRING ( SUBSTR(TRY16,1,NT) ) EDIT
 253                          (SAMP) ( F(NT) ) ;
 254                     ELSE GET STRING ( SUBSTR(TRY16,1,NT) ) EDIT
 255                          (SAMP) ( F(NT,1) ) ;
 256
 257                     GO TO PROCESS ;
 258
 259        DCL     TESTLINE CHAR (68) STATIC INIT (
 260        '** TESTLINE - XXXXXXXX.XX  XXXXXXXX.XX  XXXXXXXX.XX  XXXXXXXX.XX'
 261        ),
 262                TF1  PICTURE '999999999V.99' DEF TESTLINE POS    (15),
 263                TF2  PICTURE '999999999V.99' DEF TESTLINE POS    (29),
 264                TF3  PICTURE '999999999V.99' DEF TESTLINE POS    (43),
 265                TF4  PICTURE '999999999V.99' DEF TESTLINE POS    (57) ;
 266
 267        BUGDE:          TF1 = SAMP ;
 268                        TF2 = LLO (N) ;
 269                        TF3 = LHI (N) ;
 270                        DISPLAY ( TESTLINE ) ;
 271                        DISPLAY ( '**REP16**' || SUBSTR(REP,1,16)||'**') ;
 272                        DISPLAY ( '**TRY16**' || TRY16 || '**' ) ;
 273
 274
 275        PROCESS: IF ( K = 1 ) THEN GO TO MAKETRANS ;
 276
 277                IF ( K = 3 ) THEN
 278        DOHI:       DO ;
 279                    ROUTE = MAKETRANS ;
 280        TRYHI:      IF ( SAMP <= LHI(N) ) THEN GO TO ROUTE ;
 281                        LSAMP = LHI (N) ;
 282                        TYPLIM = 'HI' ;
 283        BUSTLIM:        IF ( J = 2 )
 284                            THEN ILIMVAL = LSAMP ;
 285                            ELSE FLIMVAL = LSAMP ;
 286                        DISPLAY ( LIMSG ) ;
 287                            GO TO NEWLOOP ;
 288                    END DOHI ;
 289
 290                IF ( K = 4 ) THEN
 291        DOLO:       DO ;
 292                    IF ( SAMP >= LLO(N) ) THEN GO TO MAKETRANS ;
 293                        LSAMP = LLO (N) ;
 294                        TYPLIM = 'LO' ;
 295                            GO TO BUSTLIM ;
 296                    END DOLO ;
 297
 298        TEST_BOTH_LIMITS:
 299                ROUTE = DOLO ;
 300                    GO TO DOHI ;
 301
 302        MAKETRANS:
 303                L = LEN (N) ;
 304                IF ( FT(N) = 1 ) THEN
 305        DOCHAR:     DO ;
 306                    SUBSTR ( RTRANS, START(N), L ) = TRY16 ;
 307                        GO TO TALLY ;
 308                    END DOCHAR ;
 309
 310                IF ( FT(N) ¬= 2 ) THEN GO TO DOFRACTION ;
 311        DOINTEGER:  PUT STRING ( SUBSTR(RTRANS, START(N), L) ) EDIT
 312                        (SAMP) ( F(L) ) ;
 313                        GO TO TALLY ;
 314
 315        DOFRACTION:
 316                FF = FORMS ( FSEL(N) ) ;
 317                PUT STRING ( SUBSTR(RTRANS, START(N), L) ) EDIT
 318                        (SAMP) ( R(FF) ) ;
 319                        GO TO TALLY ;
 320
 321        /*** CENTRAL TALLY SECTION FOLLOWS ***/
 322
 323        TALLY:
 324                IF ( ¬NEW ) THEN GO TO OLDE ;
 325                N = N + 1 ;
 326                IF ( N<= NMAX ) THEN GO TO NEWLOOP ;
 327
 328        FINISH_TRANSACTION:
 329                REWRITE FILE (TRANS) FROM (RTRANS) ;
 330
 331        FIN_TRAN:
 332                CLOSE FILE (TRANS) ;
 333                DISPLAY ( LBLANK ) ;
```

```
CARD NO      ****                    CONTENTS                                  ****

                 DISPLAY ( '* TRANSACTION(S) HAS/HAVE BEEN REWRITTEN' ) ;
                      NN = 0 ;
                      GO TO SETCODE ;

            TRYSET: NN = 0 ;
                 IF ( TRYS = 'DONE' ) THEN GO TO SETCODE ;

            /* TEMPORARY REPEAT FOR DEBUG PURPOSES */
                 DISPLAY ( LBLANK ) ;
                 DISPLAY ( '* DEBUG REPEATING TO LABEL = WHICHWAY *' ) ;
                      GO TO WHICHWAY ;

            /*** CORRECT AN EXISTING TRANSACTION IN THIS SECTION ***/

            CHANGEIT:
                 NEW = '0'B ;
                 PR = ADDR (REP) ;
                 PF = ADDR (FNME) ;

            OLDE:
            ASK2:  DISPLAY ( LBLANK ) ;
                    DISPLAY ( '* SPECIFY SHOW OR CHANGE' )
                      REPLY (REP) ;
                 IF ( TAYOUT = 'OUT' ) THEN GO TO OUTNOW ;
                 IF ( TRYS = 'DONE' ) THEN GO TO FINISH_TRANSACTION ;

                 IF ( TRYS = 'SHOW' ) THEN
            L7:       DO ;
                      J = 5 ;
            L7A:      MODE = 1 ;
                           GO TO LOOKB ;
                      END L7 ;
                 J = 2 ;
                 IF ( TRY12 = 'S ' ) THEN GO TO L7A ;

                 IF ( TRYC = 'CHANGE') THEN
            L8:       DO ;
                      J = 7 ;
            L8A:      MODE = 2 ;
                      GO TO LOOKB ;
                      END L8 ;
                 J = 2 ;
                 IF ( TRY12 = 'C ' ) THEN GO TO L8A ;

                      DISPLAY ( '* FIRST WORD OF REPLY UNKNOWN *' ) ;
                           GO TO ASK2 ;

            LOOKB:  IF ( REPTAB(J) ¬= ' ' ) THEN GO TO STEP5 ;

                 IF ( J < 15 ) THEN GO TO STEP4 ;
                      DISPLAY ( '* CANNOT DETERMINE ENTRY FORMAT *' ) ;
                           GO TO ASK2 ;

            STEP4:  J = J + 1 ;
                      GO TO LOOKB ;
            STEP5:  FNME = ' ' ;
                 L = 1 ;
                      GO TO FILLJ ;

            FILLN:  IF ( REPTAB(J) = ' ' ) | ( REPTAB(J) = '=' )
                      THEN GO TO FIND ;
            FILLJ:  FN12 (L) = REPTAB (J) ;
                 L = L + 1 ;
                 IF ( L <= 12 ) THEN GO TO BUMPJ ;
                      DISPLAY ( '* MNEMONIC HAS TOO MANY CHARACTERS *' ) ;
                           GO TO ASK2 ;
            BUMPJ:  J = J + 1 ;
                      GO TO FILLN ;

            FIND:     DO K = 1 TO NMAX ;
                      N = K ;
                      IF ( FNME = MN(N) ) THEN GO TO STEP6 ;
                      END FIND ;

                           DISPLAY ( '* UNKNOWN MNEMONIC - TRY AGAIN *' ) ;
                      GO TO ASK2 ;

            STEP6:  IF ( MODE = 2 ) THEN GO TO STEP7 ;
                 FILLV = SUBSTR (RTP.NS, START'J), LEN(N) ) ;
                 DISPLAY ( VMSG ) ;
                      GO TO ASK2 ;

            STEP7:  IF ( REPTAB(J) = '=' ) THEN GO TO STEP9 ;
            IF ( REPTAB(J) = ' ' ) THEN GO TO STEP8 ;

            STEP7A:      DISPLAY ( '* ENTRY FORMAT ?????? - TRY AGAIN *' ) ;
                      GO TO ASK2 ;

            STEP8:  IF ( J >= 80 ) THEN GO TO STEP7A ;
                 J = J + 1 ;
                      GO TO STEP7 ;

            STEP9:  IF ( J > 79 ) THEN GO TO STEP7A ;
            STEP9A: J = J + 1 ;
                 IF ( REPTAB(J) ¬= ' ' ) THEN GO TO STEP10 ;
                 IF ( J = 80 ) THEN GO TO STEP7A ;
                      GO TO STEP9A ;

            STEP10: TRY16 = SUBSTR ( REP, J ) ;
                      GO TO CHARACTER_FIELD ;

            /*** ERROR TERMINATIONS HERE ***/

            BUST:
                 NNER = NER ;
                 DISPLAY ( ERMSG ) ;

            /*** TERMINUS SECTION FOLLOWS ***/

            OUTNOW:
                 NN = 4 ;
                 CLOSE FILE (TRANS) ;
```

INPUT LISTING AUTOFLOW CHART SET - FINAL PL/1 -PROMPTX- FLOWCHARTS

CARD NO **** CONTENTS ****

```
                DISPLAY ( '* NOW EXITING BY COMMAND *' ) ;                    00044500
                DISPLAY ( '*     TRANSACTION HAS NOT BEEN RE-WRITTEN *' ) ;   00044600
                                                                              00044700
        SETCODE:     CALL IHESARC (NN) ;                                      00044800
                                                                              00044900
                     END PROMPTX ;                                            00045000
```

CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

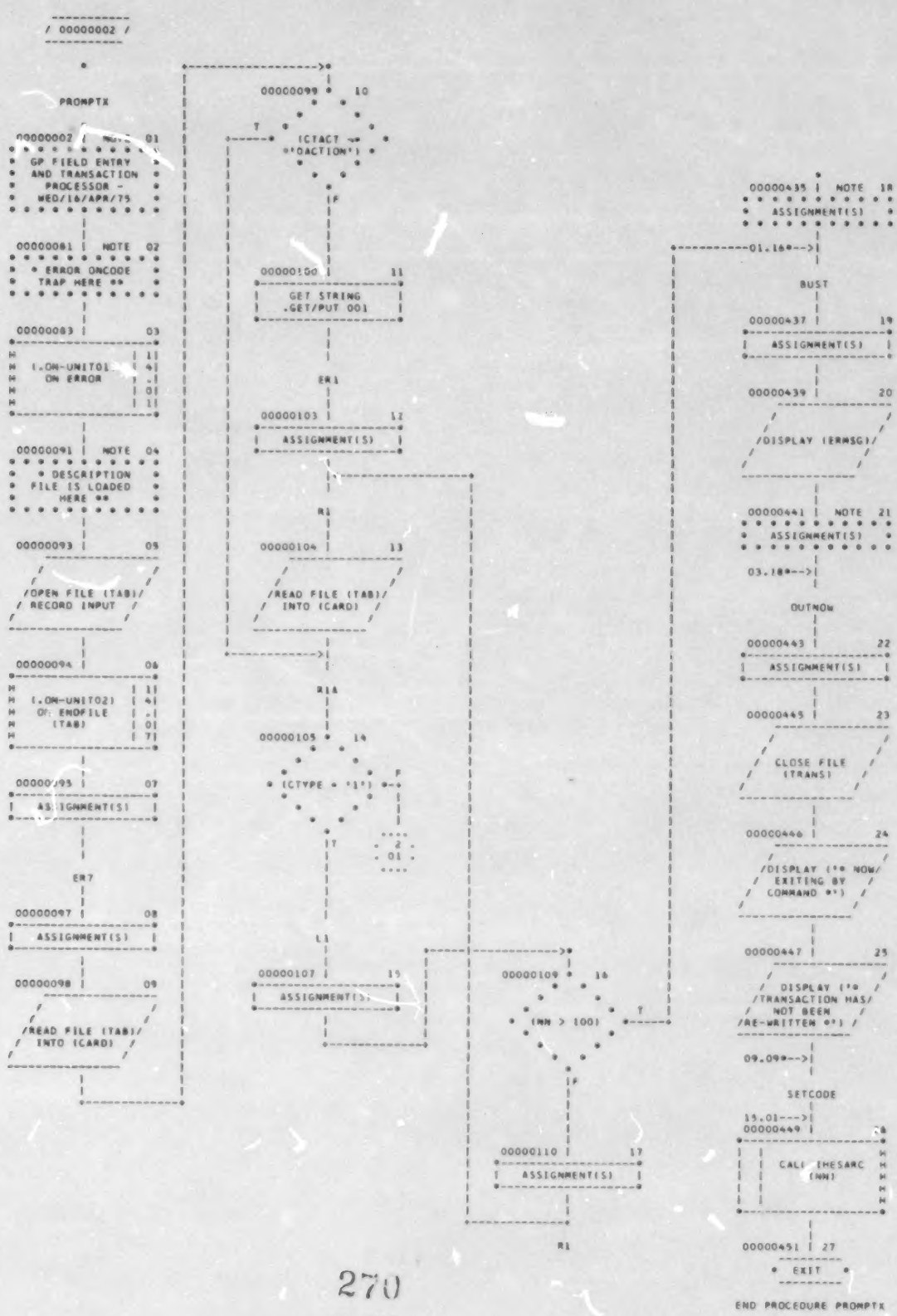


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

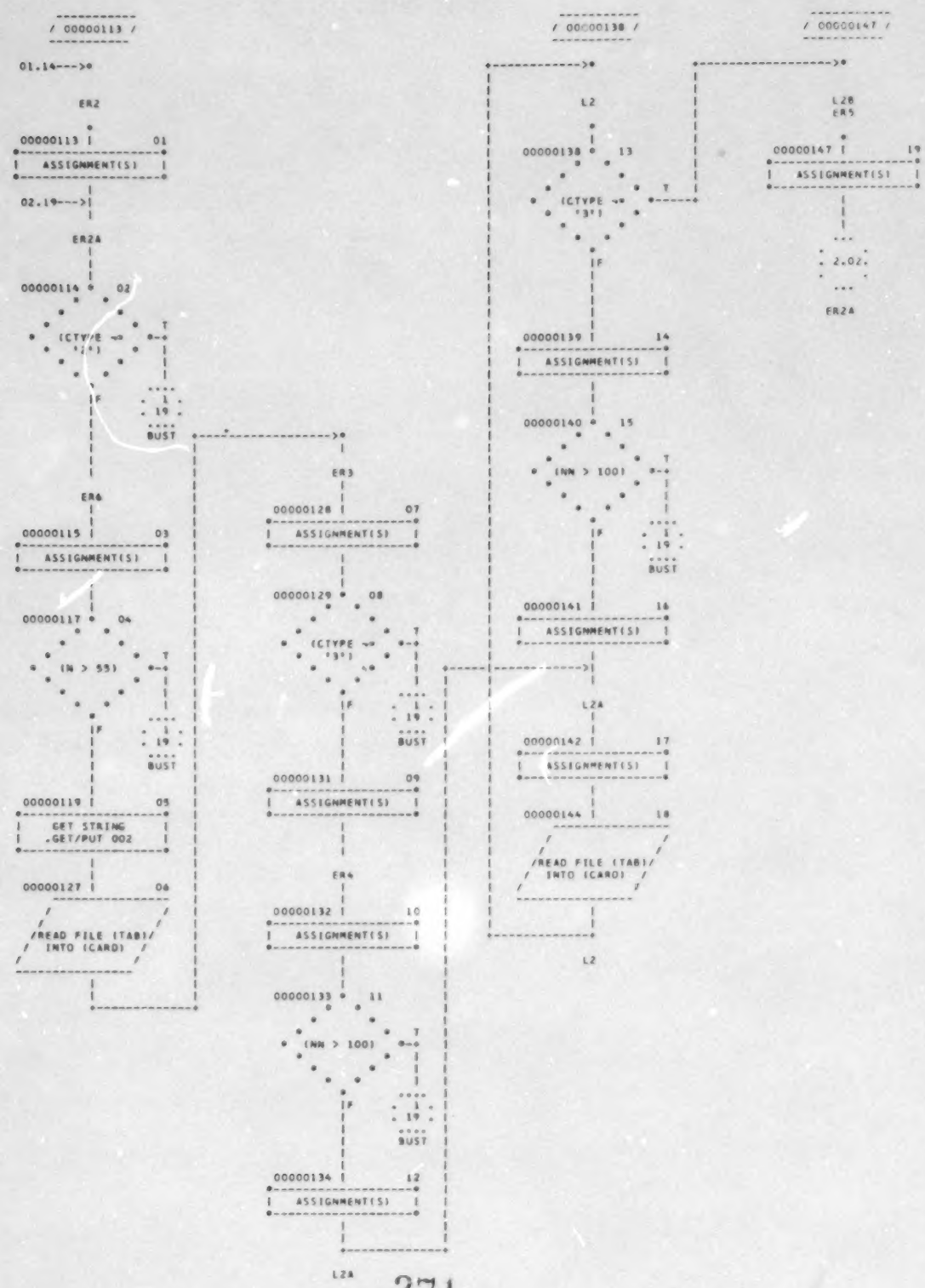


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

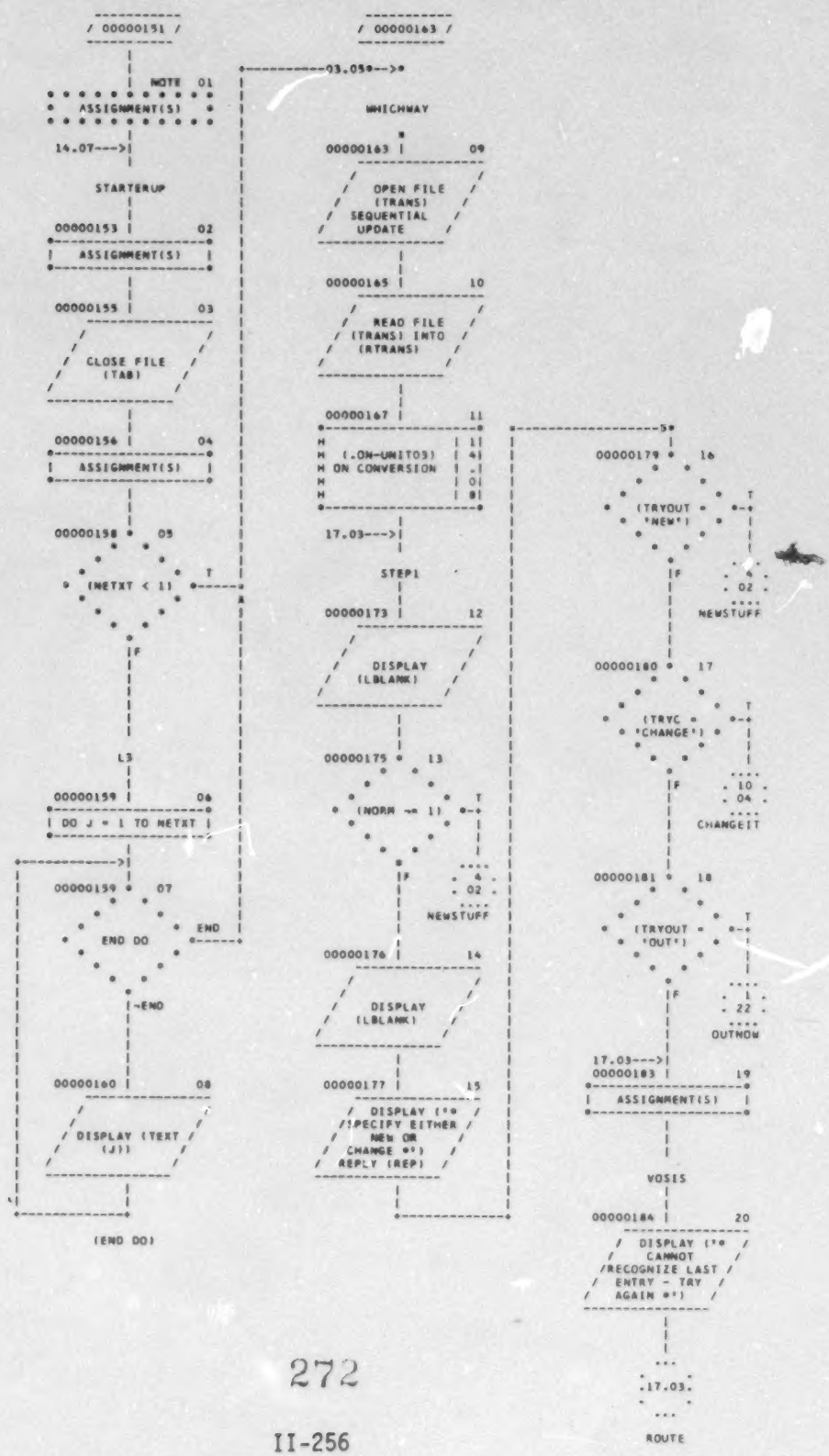


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

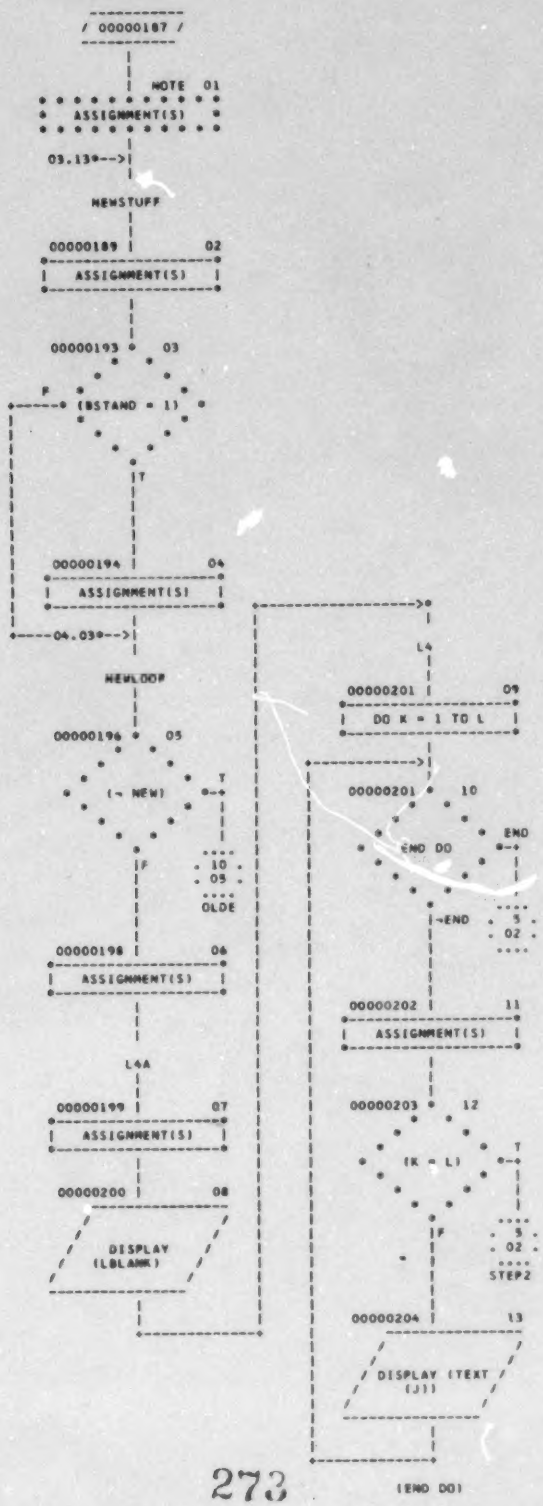


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

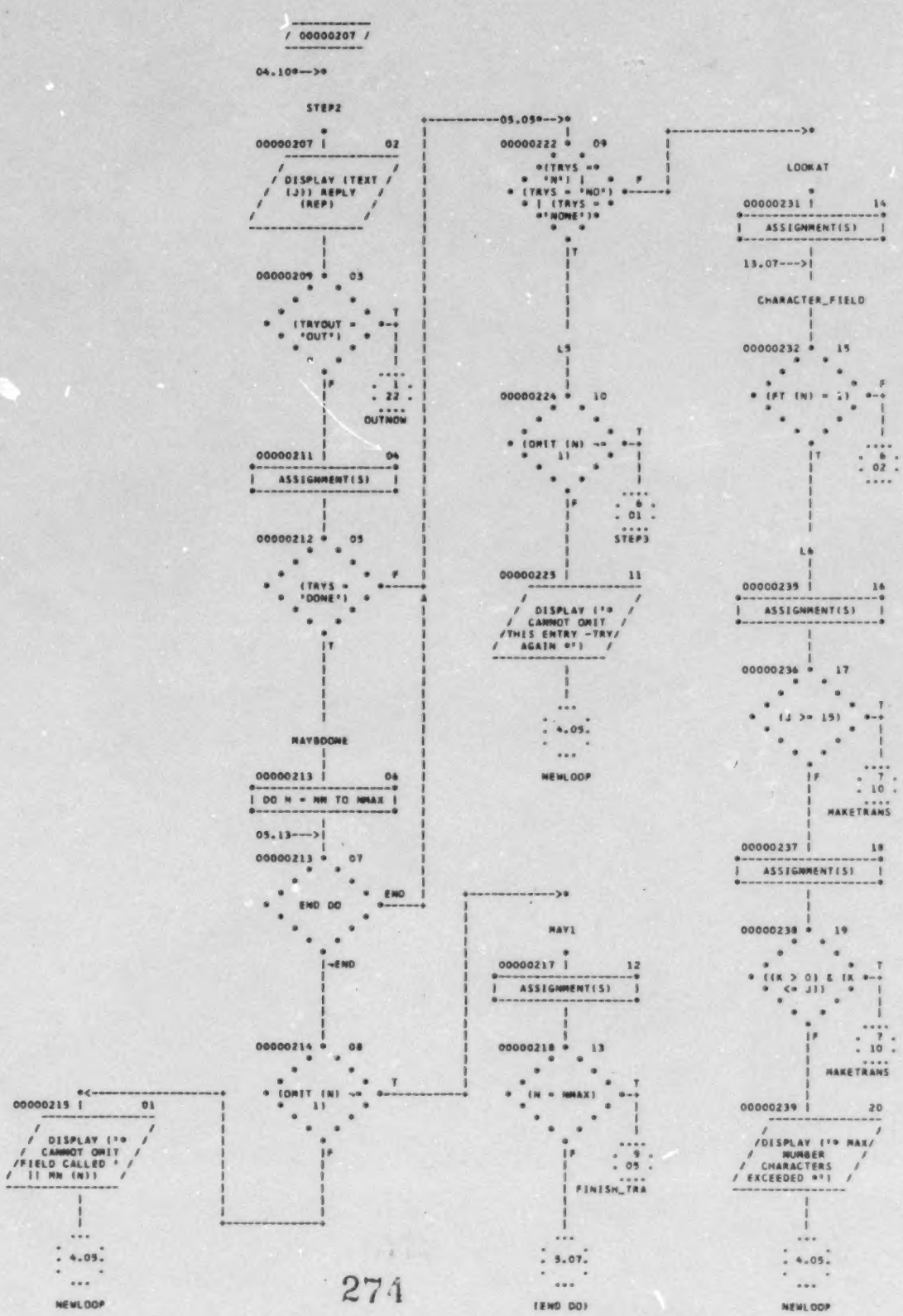


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

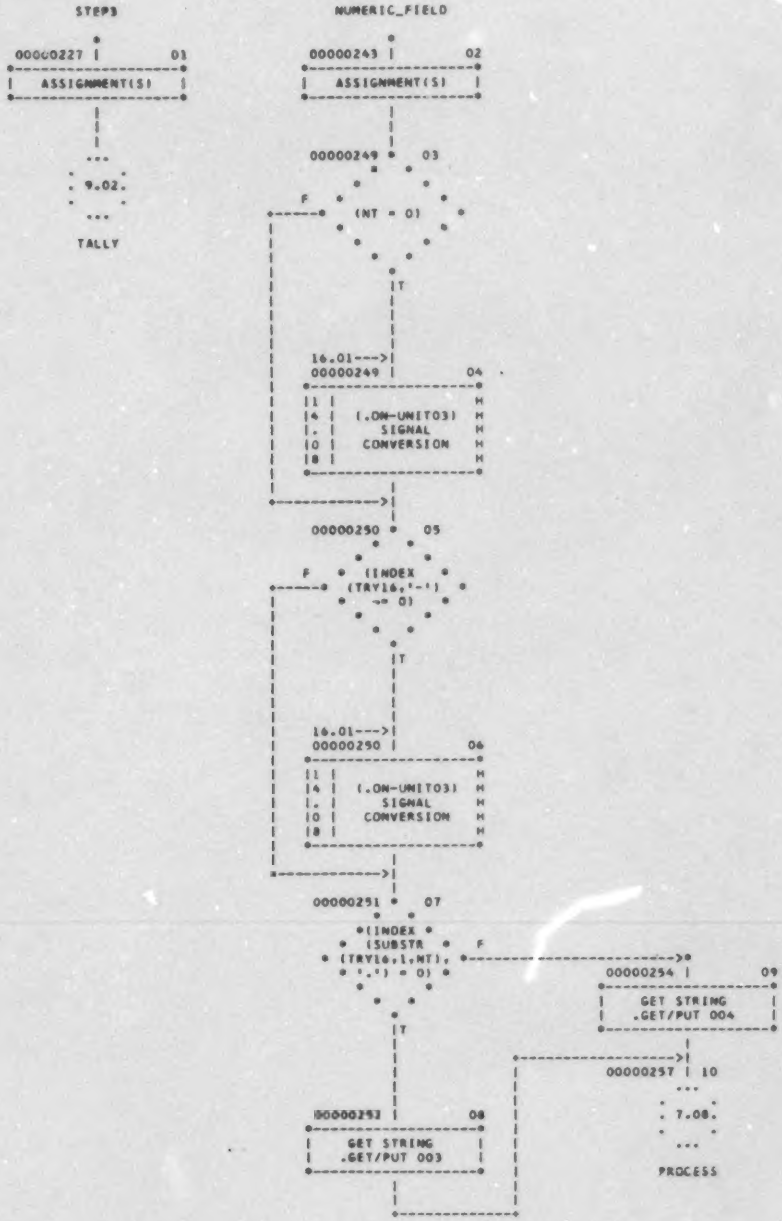


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

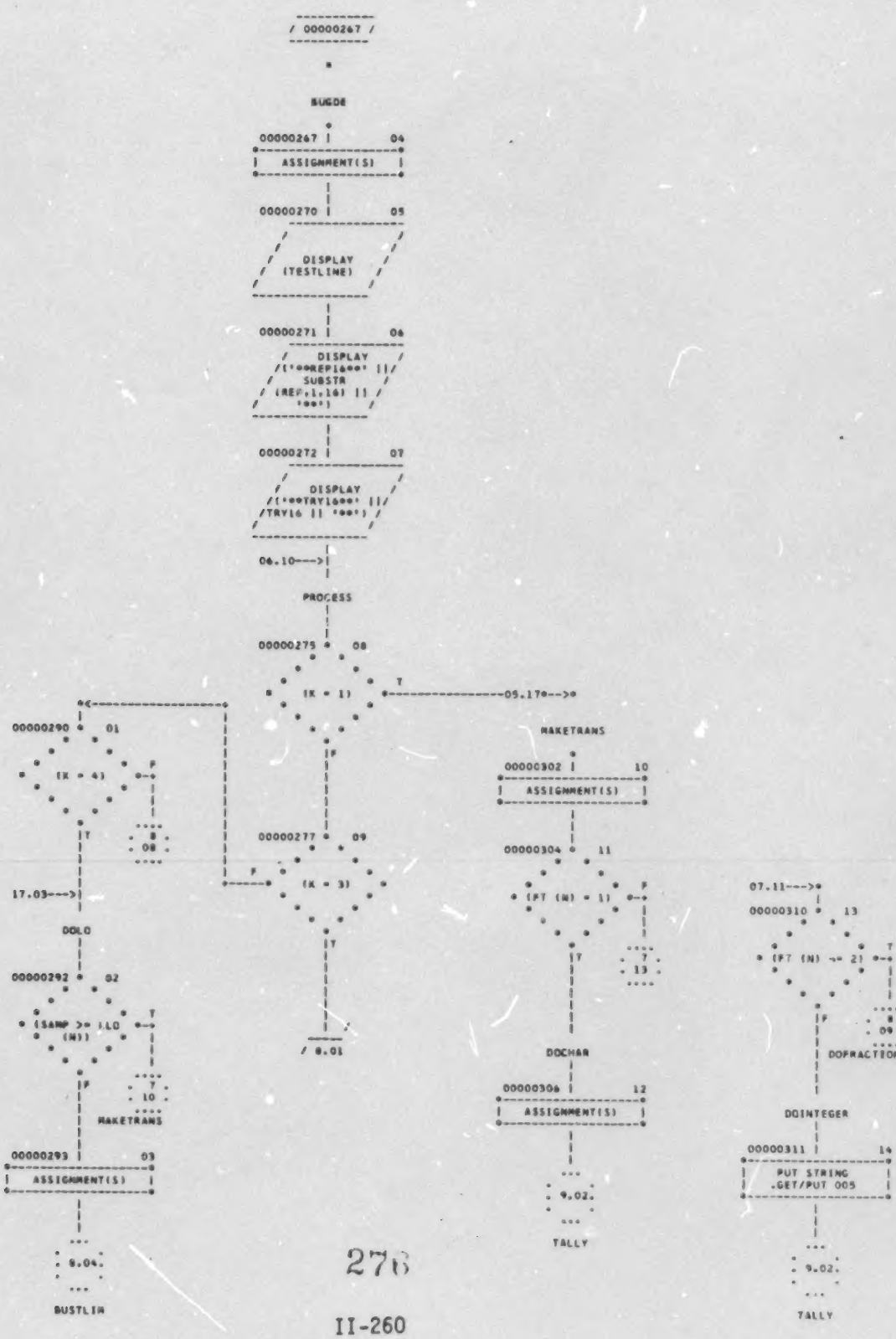


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

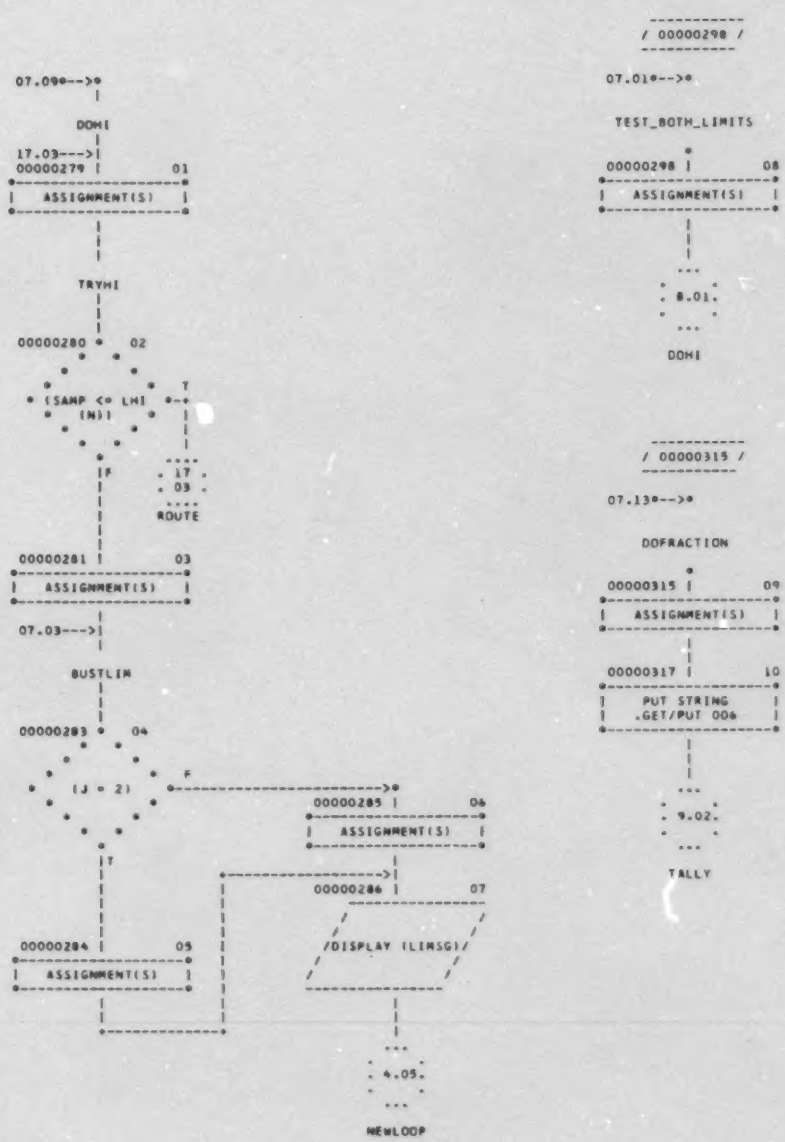
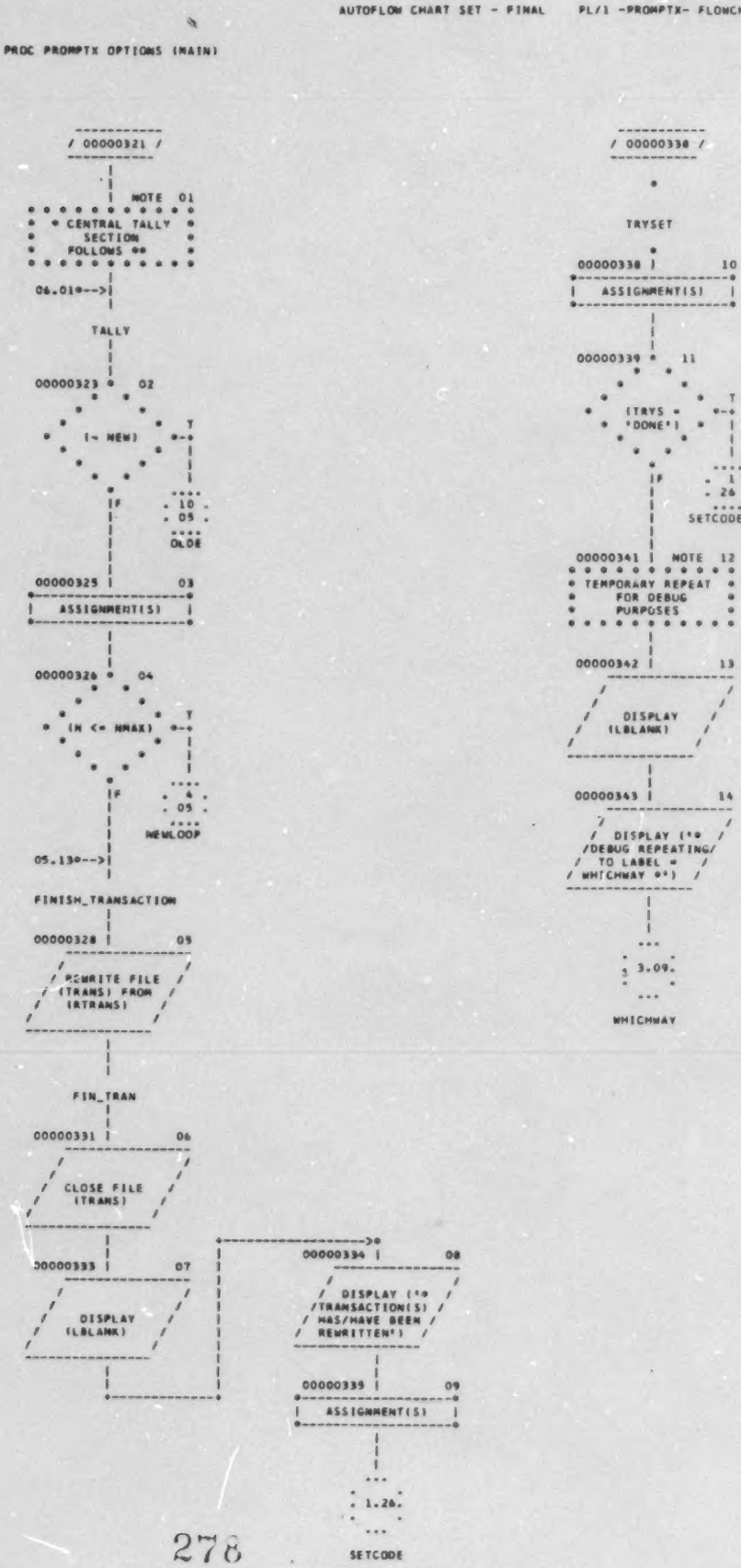


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)



278

CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

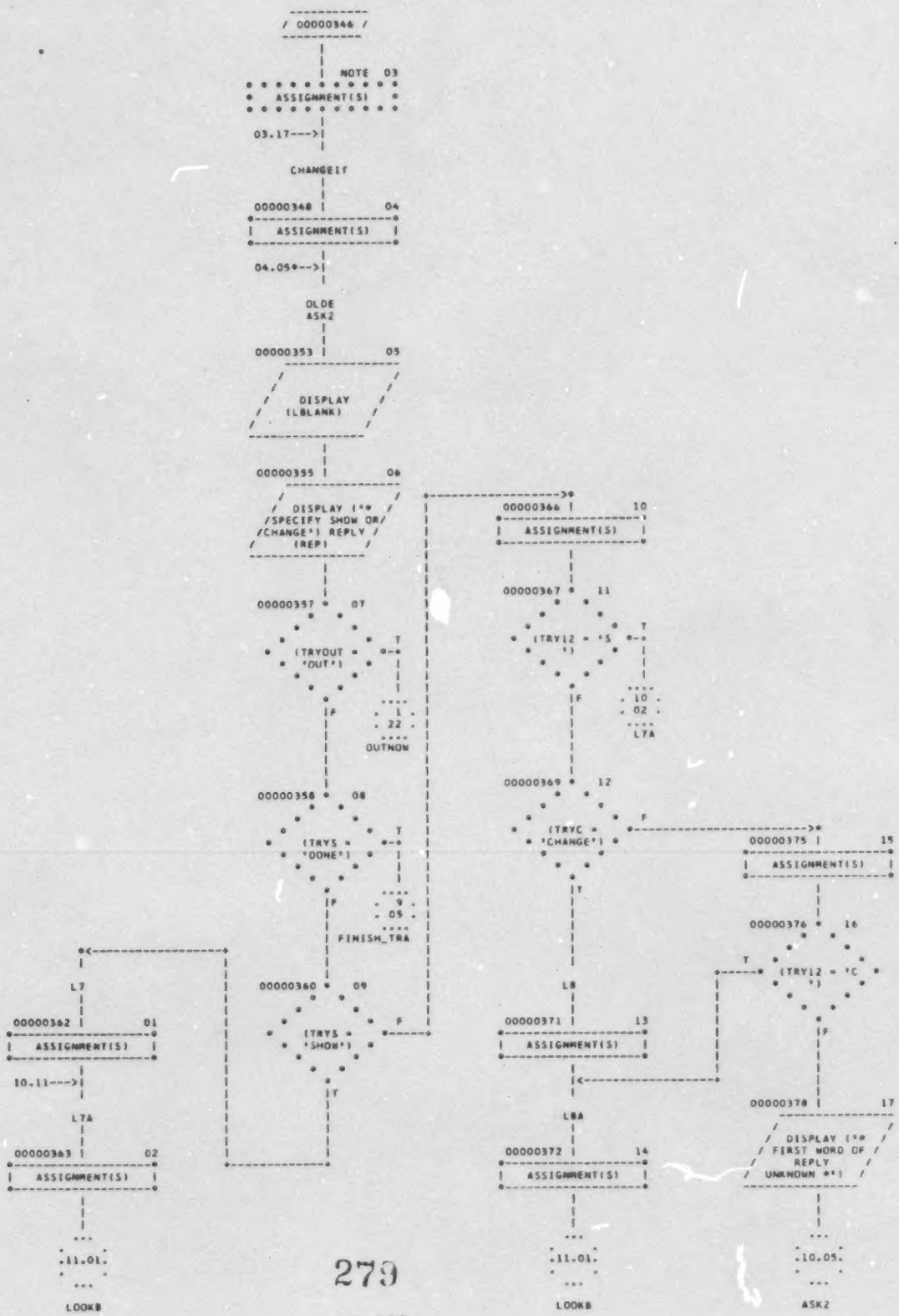


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

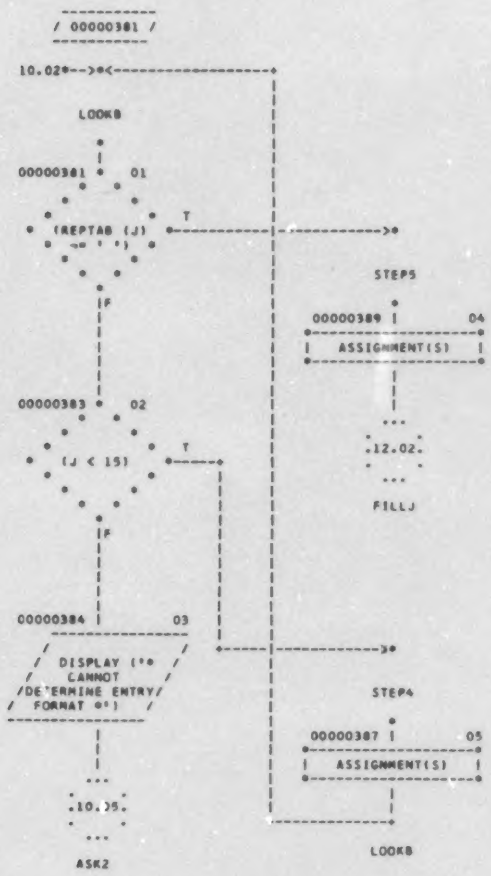


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

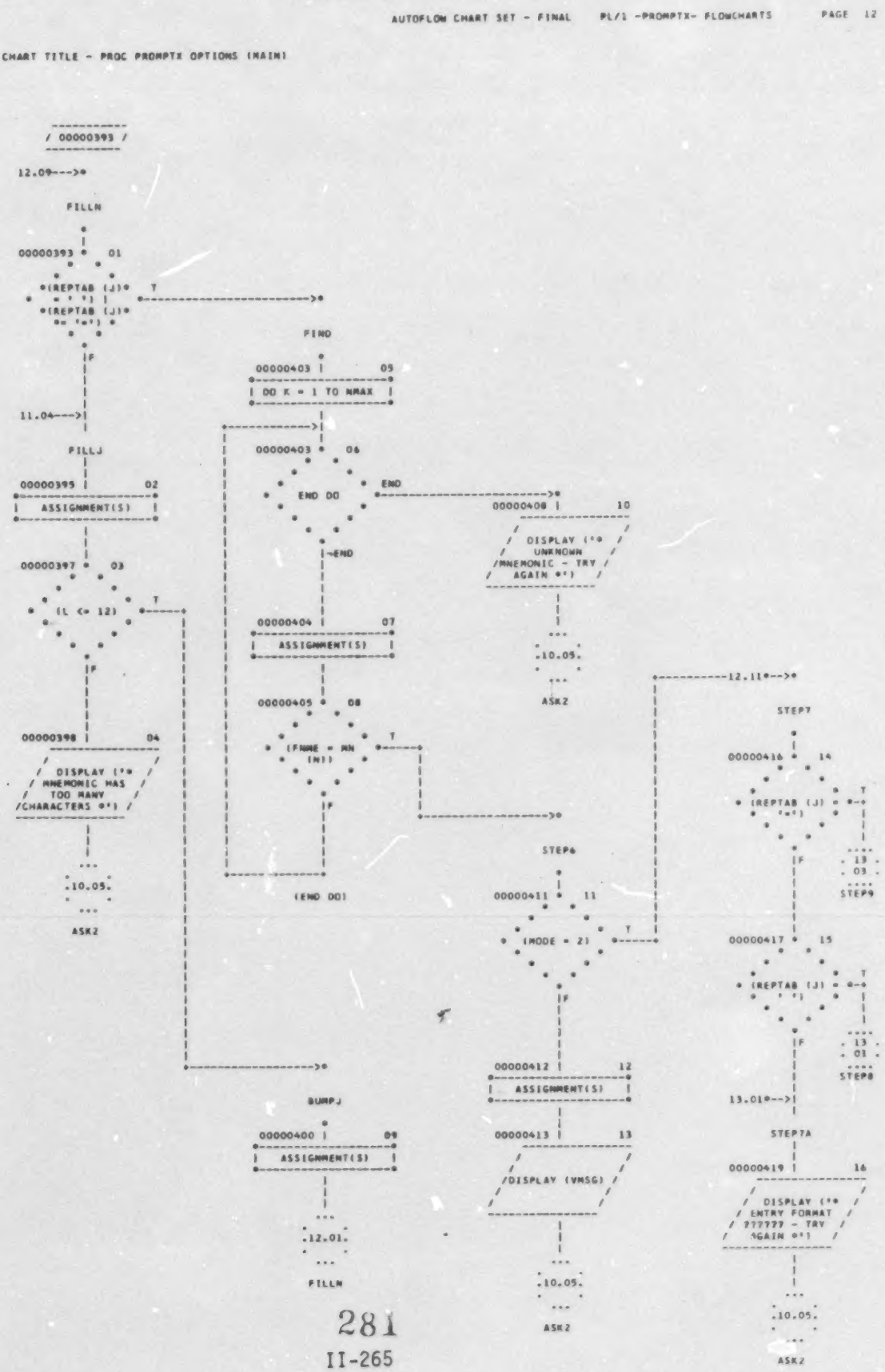


CHART TITLE - PROC PROMPTX OPTIONS (MAIN)

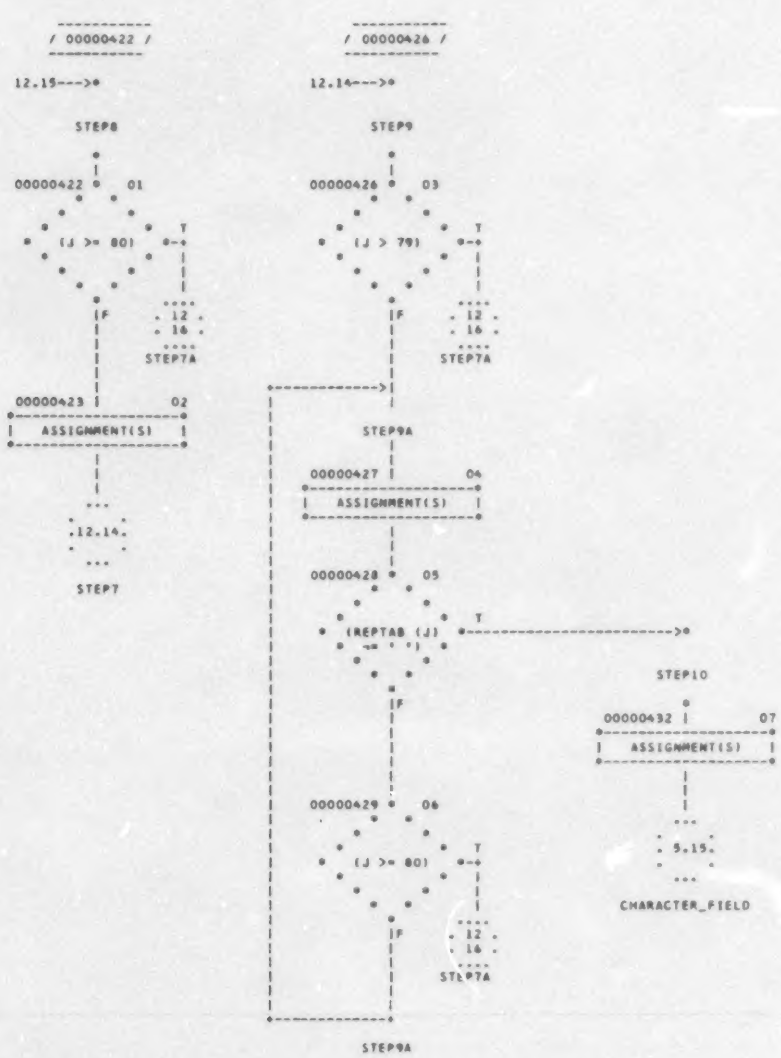


CHART TITLE - ON-UNIT ACTION BLOCKS

```
          ----------
         / 00000083 /
         ----------

      01.03--->*

      .ON-UNIT01

                *
      00000083 |  NOTE  01
      * * * * * * * * * * *
      *     ON ERROR      *
      *                   *
      *       BEGIN       *
      * * * * * * * * * * *
                |
                |
      00000084 |        02
      *-------------------*
      |   ASSIGNMENT(S)   |
      *-------------------*
                |
                |
      00000085 |        03
          ----------------
         /                /
        /                /
       /DISPLAY (DERM) /
      /               /
     /               /
     ----------------
                |
                |
                |
      00000086 *   04
              *   *
            *       *
      F   *           *
   +-----*   (FIRST)    *
   |       *           *
   |         *       *
   |           *   *
   |             *
   |             |T
   |             |
   |             |
   |             |
   |             |
   |             |
   |   00000086 |        05
   |       ----------------
   |      /                /
   |     /                /
   |    / DISPLAY (F70) /
   |   /               /
   |  /               /
   |  ----------------
   |             |
   +------------>|
      00000087 |        06
      *-------------------*
      |   ASSIGNMENT(S)   |
      *-------------------*
                |
                |
                |
               ...
             .     .
             . 1.26.
             .     .
               ...

            SETCODE

         END .ON-UNIT01
```

```
          ----------
         / 00000167 /
         ----------

      03.11*-->*

      .ON-UNIT03

                *
      00000167 |  NOTE  08
      * * * * * * * * * * *
      *   ON CONVERSION   *
      *                   *
      *       BEGIN       *
      * * * * * * * * * * *
                |
                |
      00000168 |        09
      *-------------------*
      |   ASSIGNMENT(S)   |
      *-------------------*
                |
                |
      00000169 |        10
          ----------------
         /                /
        /                /
       /DISPLAY (DERM) /
      /               /
     /               /
     ----------------
                |
                |
                |
               ...
             .     .
             . 4.05.
             .     .
               ...

            NEWLOOP

         END .ON-UNIT03
```

```
          ----------
         / 00000094 /
         ----------

      01.06--->*

      .ON-UNIT02

                *
      00000094 |  NOTE  07
      * * * * * * * * * * *
      * ON ENDFILE (TAB)  *
      * * * * * * * * * * *
                |
                |
                |
               ...
             .     .
             . 3.02.
             .     .
               ...

            STARTERUP

         END .ON-UNIT02
```

283

CHART TITLE - CALLED PROCEDURES - CROSS REFERENCE

```
          ENTRY
            *
            |          01
*-------------------*
H                   |  |
H    IHESARC        |  |
H  CALLED FROM      |  |
H                   |  |
H                   |  |
*-------------------*
|  .10003       1.26 |
*-------------------*
```

CHART TITLE - SIGNALLED ON-UNIT ACTION BLOCKS

CONVERSION

ON-UNIT BLOCK
.ON-UNIT03 14.08
SIGNALLED AT
.10001 4.06
.10002 4.06

CHART TITLE - LABEL ASSIGNMENT CROSS REFERENCE

```
      -----------
     /    FF    /
     -----------
          |
          |
          |          01
*-------------------*
|      SCALAR       |
|  LABEL VARIABLE   |
*-------------------*
| SET \T  TO GO TO  |
|    8.09   17.02   |
| 00000316 FORMS    |
*-------------------*
```

```
      -----------
     /  FORMS   /
     -----------
          |
 17.01--->|
          |          02
*-------------------*
|       ARRAY       |
|  LABEL VARIABLE   |
*-------------------*
|  VALUE AMBIGUOUS  |
*-------------------*
```

```
      -----------
     /  ROUTE   /
     -----------
          |
 03.20*-->|
          |          03
*-------------------*
|      SCALAR       |
|  LABEL VARIABLE   |
*-------------------*
| SET AT  TO GO TO  |
|    3.19    3.12   |
| 00000183 STEP1    |
|    8.01    7.10   |
| 00000279 MAKETRAN |
|    8.08    7.02   |
| 00000299 DOLO     |
*-------------------*
```

CHART TITLE - GET/PUT, FORMAT STATEMENTS

```
00000075 RF1        FORMAT (F (6,1))
00000076 RF2        FORMAT (F (5,1))
00000077 RF3        FORMAT (F (3,1))
00000078 RF4        FORMAT (F (4,1))
00000079 RF5        FORMAT (F (5,3))
00000100             .GET/PUT 001  GET STRING (CARD) EDIT (NORM,BSTAND,ASTAND) (X (9),F (1),F (2),F (2))
00000119             .GET/PUT 002  GET STRING (CARD) EDIT (MN (N),F0 (N),FT (N),MAXC (N),OSS (N),DEF (N),OMIT (N),
00000121                       LIMIT (N),LLO (N),LHI (N),FSEL (N),START (N),LEN (N),NLINE (N),LISS (N)) (X (1),A (10),
00000123                       F (2),F (1),F (2),F (2),A (16),F (1),F (1),F (10,2),F (10,2),F (2),F (3),F (2),F (1),
00000125                       F (2))
00000252             .GET/PUT 003  GET STRING (SUBSTR (TRY16,1,NT)) EDIT (SAMP) (F (NT))
00000254             .GET/PUT 004  GET STRING (SUBSTR (TRY16,1,NT)) EDIT (SAMP) (F (NT,1))
00000311             .GET/PUT 005  PUT STRING (SUBSTR (RTRANS,START (N),L)) EDIT (SAMP) (F (L))
00000317             .GET/PUT 006  PUT STRING (SUBSTR (RTRANS,START (N),L)) EDIT (SAMP) (R (PF))
```

SECTION III

ETE MODEL

MODEL DESCRIPTION

FUNCTIONAL DESCRIPTION. The ETE Model is a set of six (6) NCSS Executive procedures, two (2) FORTRAN programs and one GPSS program. During operational use, the system will also include application-specific GPSS programs generated by the user. The basic purpose of this set of procedures and programs is either to: produce a new, executable GPSS model or update an existing GPSS model. In addition to the programs already mentioned, the ETE model makes extensive use of the NCSS edit program in several modes of operation.

Figure III-1 shows the different possible execution modes of the ETE model. Figure III-2 is a more detailed presentation of overall system logic. The name of the procedure or program used to execute each step in system operation is shown in Figure III-2 along with a few explanatory words about some of the procedures and programs. Table III-1 lists the names of all procedures and programs with a brief description of each one.

The ETE model uses all standard NCSS conventions for file specification and usage. Terminal input is FORTRAN logical 5 and output is FORTRAN logical 6. File descriptions in terms of record length, sequential/random etc. can be obtained by checking the filetype designation against the file descriptions contained in the NCSS documentation. Data files used by the ETE model are summarized in Figure III-3.

Throughout this section, a procedure or executive always refers to a control element written in the NCSS EXEC or CSS languages. A program will always be written in either FORTRAN or GPSS. Note that, as delivered, no executable GPSS programs (models) exist. In order to test the system, an executable GPSS model must be developed, following standard procedures, and that model executed (see User's Guide).

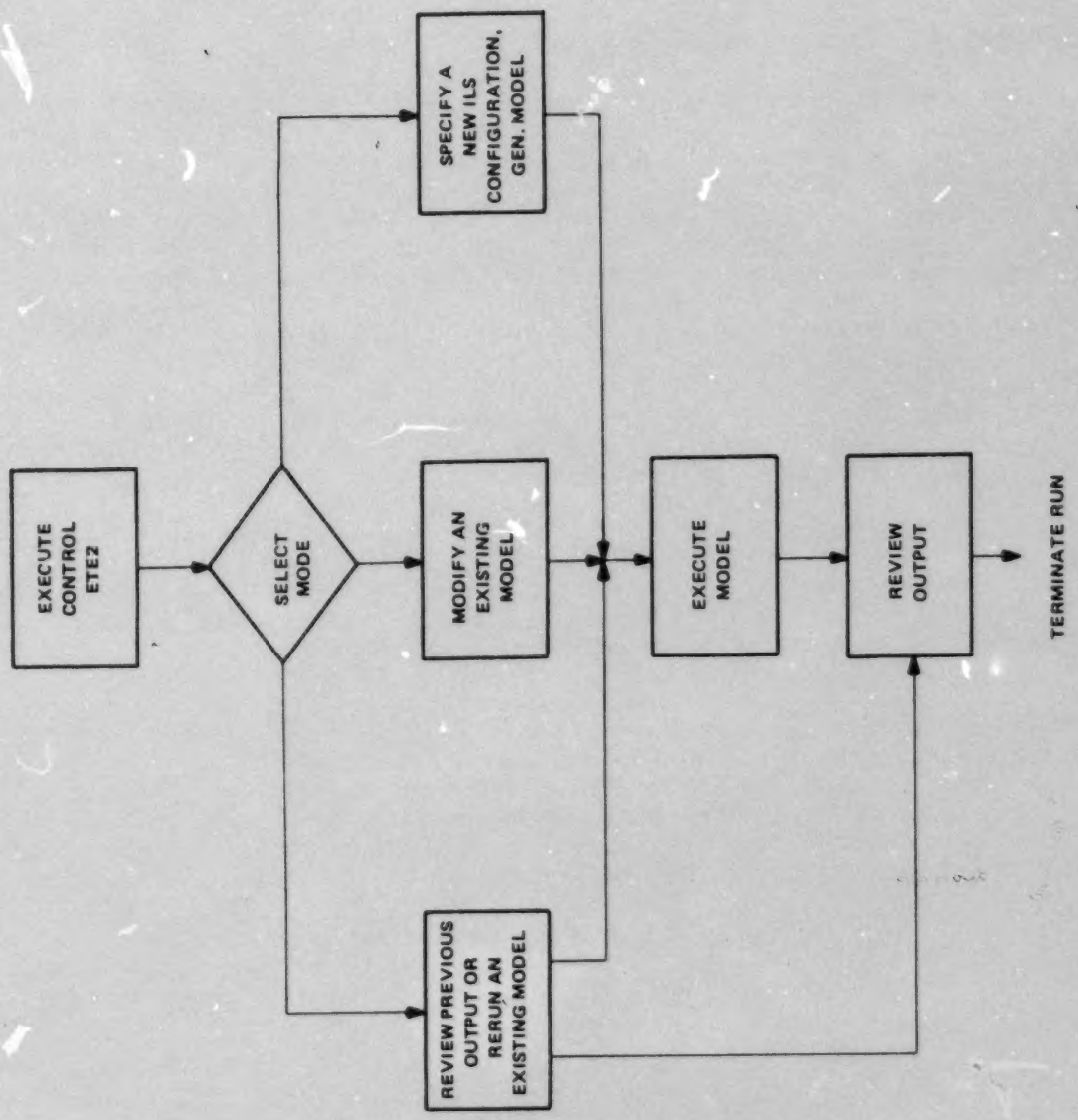
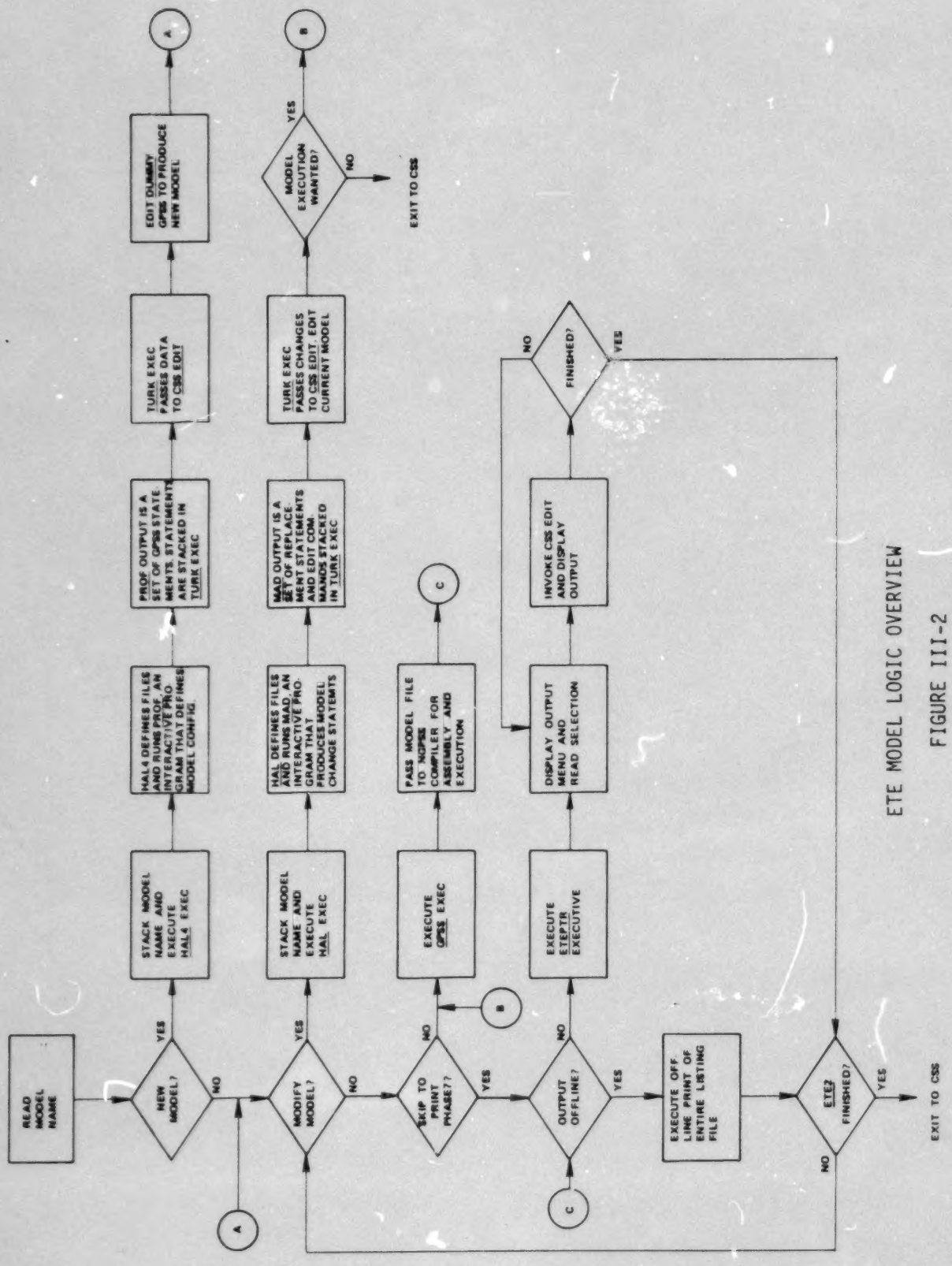


FIGURE III-1 ETE MODEL OPERATION PROCEDURE OVERVIEW



ETE MODEL LOGIC OVERVIEW

FIGURE III-2

ETE2 - The main control executive for the system. ETE2 is always invoked to begin an operational session and is returned to after each step.

HAL4 - Executive - Defines a file called TURK EXEC as logical unit 3 and causes execution of program PROF.

PROF - Program - A FORTRAN program which interacts with the user in in conversational mode and generates a set of GPSS source statements. These source statements are written on logical 3 (TURK EXEC)

TURK - Executive - a skeleton executive which edits either original GPSS source statements into DUMMY GPSS on change statements into an existing GPSS file named by the user.

DUMMY - Program - a GPSS program containing all model execution logic but no data. Execution data (i.e., ILS configuration specifications) are entered by TURK EXEC in the form of GPSS source statements (FUNCTION statements, VARIABLE statements, etc.).

HAL - Executive - This executive defines a file (Logical 3) which carries the name TURK EXEC and executes program MAD.

MAD - Program - a FORTRAN program which accepts model change requests from the user and generates the necessary replacement GPSS statements or CDS EDIT commands required to incorporate these changes in the model file specified by the user. MAD uses logical 2 as the existing model file and logical 3 as the file on which the change requests are listed.

ETEPTR - Executive - This executive locates the output listing of the current model execution run and prints or displays it. The output file may be listed in its entirety or in parts according to user request.

GPSS - Executive - GPSS EXEC stacks answers to all NGPSSV messages and passes the current model file name to the NGPSSV system for compilation and execution.

ETE PROGRAMS AND EXECUTIVES

TABLE III-1

FILE NAME	FILE TYPE	FORTRAN UNIT	PROGRAM USING	CONTENTS
USER SUPPLIED (&MODEL)	GPSS	2	MAD NGPSSV	User Generated GPSS Source Programs.
TURK	EXEC	3	PROF MAD	CSS EDIT Commands for Model Generation and Update
DUMMY	GPSS	-	TURK	GPSS Source Statements
&MODEL (USER SUPPLIED)	LISTING	-	ETEPTR	Output from Model Execution

ETE DATA FILE SUMMARY

FIGURE III-3

EXECUTIVE SUBSYSTEM

Within this subsystem are all the procedures written in the NCSS EXEC or CSS languages. All control of program execution is contained within this subsystem as are all provisions for user selection of different system paths. Two of the six executives exist as separate procedures only because they are sometimes used separately during debugging of the two major FORTRAN programs in the system. TURK EXEC is a skeleton procedure which is generated for each execution. Because of this, only two procedures: ETE2 and ETEPTR will be discussed in detail and flow diagrams provided. Figure III-4 is a flow diagram of the executive subsystem.

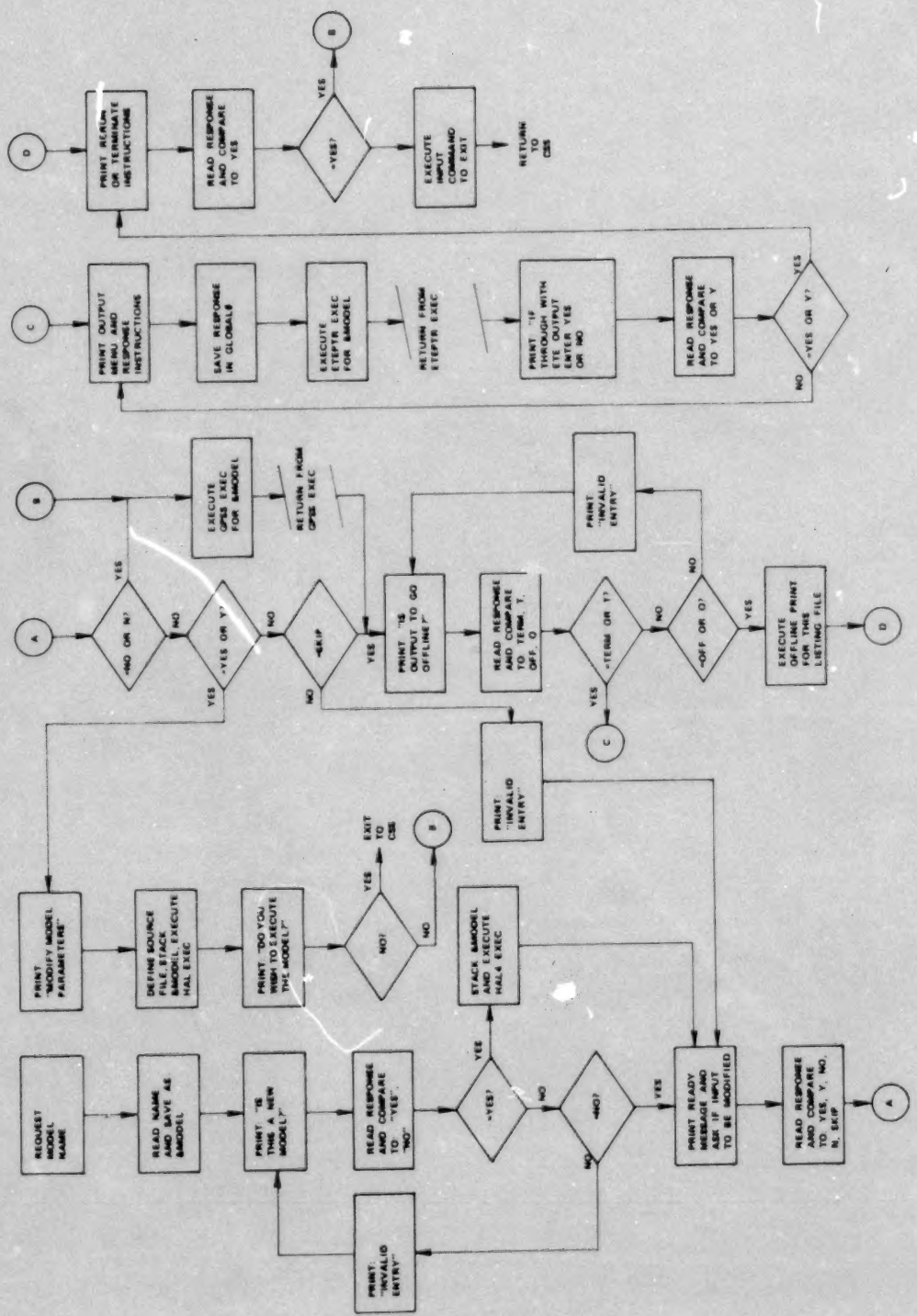
ETE2 is the control executive for the complete ETE model system. All returns from other executives and programs are made either to ETE2 or to CSS control.

The ETE model system is invoked by entering: ETE2 at the terminal. Once control has been passed to ETE2 the system is transparent to the user with regard to program or executive names. All control takes place via YES/NO or menu selection (integer) response.

Currently, no provision is made for the user to force exit from ETE2, once it has been entered, other than the standard CSS usage of the BREAK or ATTN keys. This procedure is outlined in the NCSS VP/CSS Reference Manual.

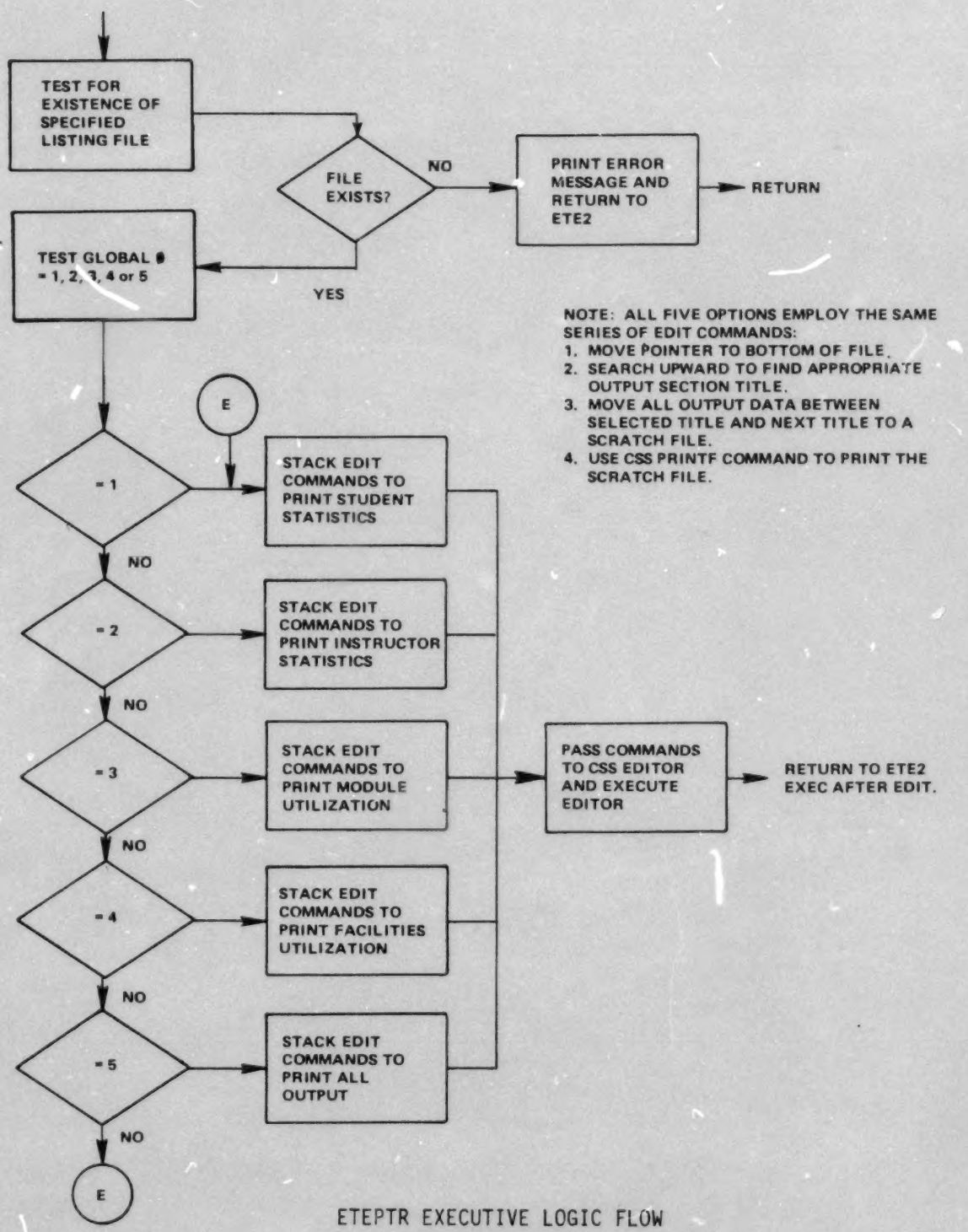
ETE2 control flow is presented in macro form in Figure III-1 and in more detailed form (including program and procedure names) in Figure III-2. Figure III-4 contains detailed logic flows of the two principal executives ETE2 and ETEPTR.

ETEPTR depends upon the presence of a LISTING file formatted by the report writer contained in DUMMY GPSS. ETEPTR searches for the output section headings supplied by the report writer (e.g. STUDENT STATISTICS, INSTRUCTOR STATISTICS, etc.) and will not provide output from a standard GPSS output format. In the event that model execution terminates prematurely for any reason (and the report writer is not entered) ETEPTR cannot be used for output display.



ETE2 EXECUTIVE LOGIC FLOW

FIGURE III-4



ETEPTR EXECUTIVE LOGIC FLOW

FIGURE III-4

```
ETE2      EXEC
         VP/CSS


&TYPE OFF
-LABA &PRINT PLEASE ENTER THE NAME OF YOUR MODEL.
&READ ARGS
&IF  &INDEX EQ 0 &GOTO -LABA
&MODEL = &1
-LABB &PRINT IS THIS A NEW MODEL?
&READ ARGS
&IF &INDEX EQ 0 &GOTO -LABB
&IF &1 EQ YES &GOTO -LABC
&IF &1 EQ NO &GOTO -LAB0
&PRINT INVALID ENTRY
&GOTO -LABB
-LABC &STACK &MODEL
EXEC HAL4
-LAB0 &PRINT ETE EXEC READY FOR INPUTS
-LAB1 &PRINT IS ETE INPUT TO BE MODIFIED
&PRINT ENTER : YES OR NO (TO REVIEW PREVIOUS OUTPUT ENTER : SKIP)
&READ ARGS
&IF &INDEX EQ 0 &SKIP -3
&IF &1 EQ NO &GOTO -ENT2
&IF &1 EQ N &GOTO -ENT2
&IF &1 EQ YES &GOTO -ENT1
&IF &1 EQ Y &GOTO -ENT1
&IF &1 EQ SKIP &GOTO -SKIP
&PRINT INVALID ENTRY
&SKIP -9
-ENT1 &PRINT MODIFY MODEL PARAMETERS
&STACK &MODEL
FILEDEF 2 DSK &MODEL GPSS DISP OLD
EXEC HAL &MODEL
&PRINT DO YOU WISH TO EXECUTE THE MODEL?
&READ ARGS
&IF &1 EQ YES &GOTO -ENT2
&IF &1 EQ NO &EXIT
-ENT2 EXEC GPSS &MODEL
-SKIP &CONTINUE
&PRINT IS ETE OUTPUT TO GO OFF LINE
&PRINT ENTER : OFF OR TERM
&READ ARGS
&IF &INDEX EQ 0 &SKIP -3
&IF &1 EQ TERM &GOTO -LAB3
&IF &1 EQ T &GOTO -LAB3
&IF &1 EQ OFF &GOTO -LAB4
&IF &1 EQ O &GOTO -LAB4
&PRINT INVALID ENTRY
&SKIP -9
-LAB4 OFFLINE PRINT ETE LISTING
&GOTO -TERM
```

ETE EXECUTIVE LISTINGS
TABLE III-2

```
ETE2      EXEC

-LAB3 &PRINT FOLLOWING ETE DATA AVAIL.
&PRINT 1). STUDENT STATISTICS
&PRINT 2). INSTRUCTOR STATISTICS
&PRINT 3). MODULE UTILIZATION
&PRINT 4). OTHER FACIL UTILIZ
&PRINT 5). ALL OUTPUT
&PRINT ENTER 1,2,3,4,5
&READ ARGS
&IF &INDEX EQ 0 &SKIP -3
&GLOBAL0 = &1
EXEC ETEPTR &MODEL
&CONTINUE
&PRINT IF THROUGH WITH ETE OUTPUT
&PRINT ENTER : YES OR NO
&READ ARGS
&IF &INDEX EQ 0 &SKIP -3
&IF &1 EQ YES &GOTO -TERM
&IF &1 EQ Y &GOTO -TERM
&GOTO -LAB3
-TERM &PRINT TO RUN OR RERUN ETE
&PRINT ENTER : YES ELSE
&PRINT ENTER : EXIT
&READ ARGS
&IF &INDEX EQ 0 &SKIP -4
&IF &1 EQ YES &GOTO -LAB0
&1
&GOTO -LAB3
```

TABLE III-2 (CONTINUED)

```
ETEPTR   EXEC
         VP/CSS


&TYPE OFF
STATE &1 LISTING
&IF &INDEX0 EQ 1 &GOTO -ERR
&IF &GLOBAL0 EQ 1 &GOTO -PTR1
&IF &GLOBAL0 EQ 2 &GOTO -PTR2
&IF &GLOBAL0 EQ 3 &GOTO -PTR3
&IF &GLOBAL0 EQ 4 &GOTO -PTR4
&IF &GLOBAL0 EQ 5 &GOTO -PTR5
-PTR1 &BEGSTACK
BRIEF
B
UP /STUDENT STATISTICS/
VERIFY 120
PUT /INSTRUCTOR STATISTICS/
CSS PRINTF ((((()))) LISTING
QUIT
&ENDSTACK
&GOTO -COMM
-PTR2 &BEGSTACK
BRIEF
B
UP /INSTRUCTOR STATISTICS/
VERIFY 120
PUT /MODULE UTILIZATION/
CSS PRINTF ((((()))) LISTING
QUIT
&ENDSTACK
&GOTO -COMM
-PTR3 &BEGSTACK
BRIEF
B
UP /MODULE UTILIZATION/
VERIFY 120
PUT /OTHER FACILITIES/
CSS PRINTF ((((()))) LISTING
QUIT
&ENDSTACK
&GOTO -COMM
-PTR4 &BEGSTACK
BRIEF
B
```

TABLE III-2 (CONTINUED)

```
ETEPTR   EXEC


UP /OTHER FACILITIES UTIL/
VERIFY 120
PUT /FINISH/
CSS PRINTF ((((()))) LISTING
QUIT
&ENDSTACK
-COMM EDIT &1 LISTING
&EXIT
-PTR5 &BEGSTACK
BRIEF
B
UP /STUDENT STATISTICS/
VERIFY 120
PUT /FINISH/
CSS PRINTF ((((()))) LISTING
QUIT
&ENDSTACK
&GOTO -COMM
-ERR &CONTINUE
&UPRINT ETE MODEL MUST BE EXECUTED
&UPRINT FIRST BY ENTERING RUN ETE
&RETURN
```

TABLE III-2 (CONTINUED)

```
GPSS      EXEC
         VP/CSS

&TYPE OFF
&STACK N
&STACK N
&STACK N
&STACK NAME &1
&STACK OPTS B
&STACK RUN
&STACK QUIT
&STACK
ATTACH NGPSSV ALL BRIEF
SET ATTN OFF
```

```
TURK      EXEC
         VP/CSS

&TYPE OFF
&BEGSTACK
TOP
L / 1     FUNC/
R   1     FUNCTION    P9,L7
D 1
C 1,+3111
FILE FIRE     GPSS
&ENDSTACK
EDIT FIRE      GPSS
&EXIT
```

TABLE III-2 (CONTINUED)

```
HAL       EXEC
         VP/CSS

&TYPE OFF
FILEDEF 3 DSK TURK EXEC DISP NEW
RUN MAD
EXEC TURK
&EXIT
```

```
HAL3      EXEC
         VP/CSS

&TYPE OFF
FILEDEF 3 DSK TURK EXEC DISP NEW
FORTRAN PROF
RUN PROF
&EXIT
```

```
HAL4      EXEC
         VP/CSS

&TYPE OFF
FILEDEF 3 DSK TURK EXEC DISP NEW
RUN PROF
EXEC TURK
&EXIT
```

TABLE III-2 (CONTINUED)

MODEL GENERATION SUBSYSTEM

The Model Generation Subsystem consists of a single interactive FORTRAN program (PROF) which runs under control of the Executive Subsystem.

Figure III-5 is the Autoflow diagram of program PROF. However, some general statements concerning PROF will make the logic flow easier to understand.

The first READ statement in the program makes use of an NCSS program feature called stacking. When PROF is called by the ETE2 EXEC, the name of the model (and hence the name of the file where the newly generated GPSS program will be stored) is stacked in the terminal input stream via the calling sequence. Consequently, the first attempt to read logical unit 5 (the terminal) will cause the contents of the input stack to be read and no intervention on the part of the operator is required.

The GPSS function cards (see Figure III-6) generated by PROF constitute an n x m matrix where n is the number of learning modules to be simulated and m is the number of different resources which may be used in support of that module. PROF builds a similar matrix using the responses of the operator to supply values for the matrix elements. The matrix is called ITABL and is currently set at 30 X 7.

The seven columns of the matrix correspond to:

1. Table location of instructor group
2. Group size for group modules
3. Staff group number
4. Facility group number
5. Remedial module type code
6. Module type code
7. Instructor group size

In practice only that number of rows will be filled in the matrix equal to the number of learning modules in the system being simulated. Column 6 (module type code) is not part of the regular input. Its values are calculated from the other matrix elements. The module type code is a binary number that can be decoded, in the GPSS model, to determine the support requirements for the module in question.

If the column element for a particular resource is non-zero, the corresponding binary code number for that column is obtained from JTAB and added to column 6, In this way, the sum of all the individual codes for that module (row) is calculated and becomes the module type code.

Subroutine CHECK is used during the part of PROF where general model conditions, such as type of output data, are being set. Check reads each response under an "A" format, and tests the response for "YES", NO" or "QUIT". If the response is not equal to one of these, a "NOT UNDERSTOOD" message is generated and the user is queried again.

A "YES" response causes the flag word sent in the calling sequence to be set equal to 1. A "NO" response causes the flag word to be set equal to Ø. A "QUIT" response results in a call to EXIT.

The indicators set by Subroutine CHECK are:

INSF - Instructor data wanted
IFAC - Facilities data wanted
IREM - Remedial material used
ISTF - Staff personnel used
IGRU - Group modules used.

These indicators serve a double purpose. They are used to determine if data concerning these variables is to be requested and they are used to determine if dummy functions are required. Because the GPSS model will not assemble unless all required functions are present, PROF writes out dummy functions for those variables which the user does not require.

CHART TITLE - PROCEDURES

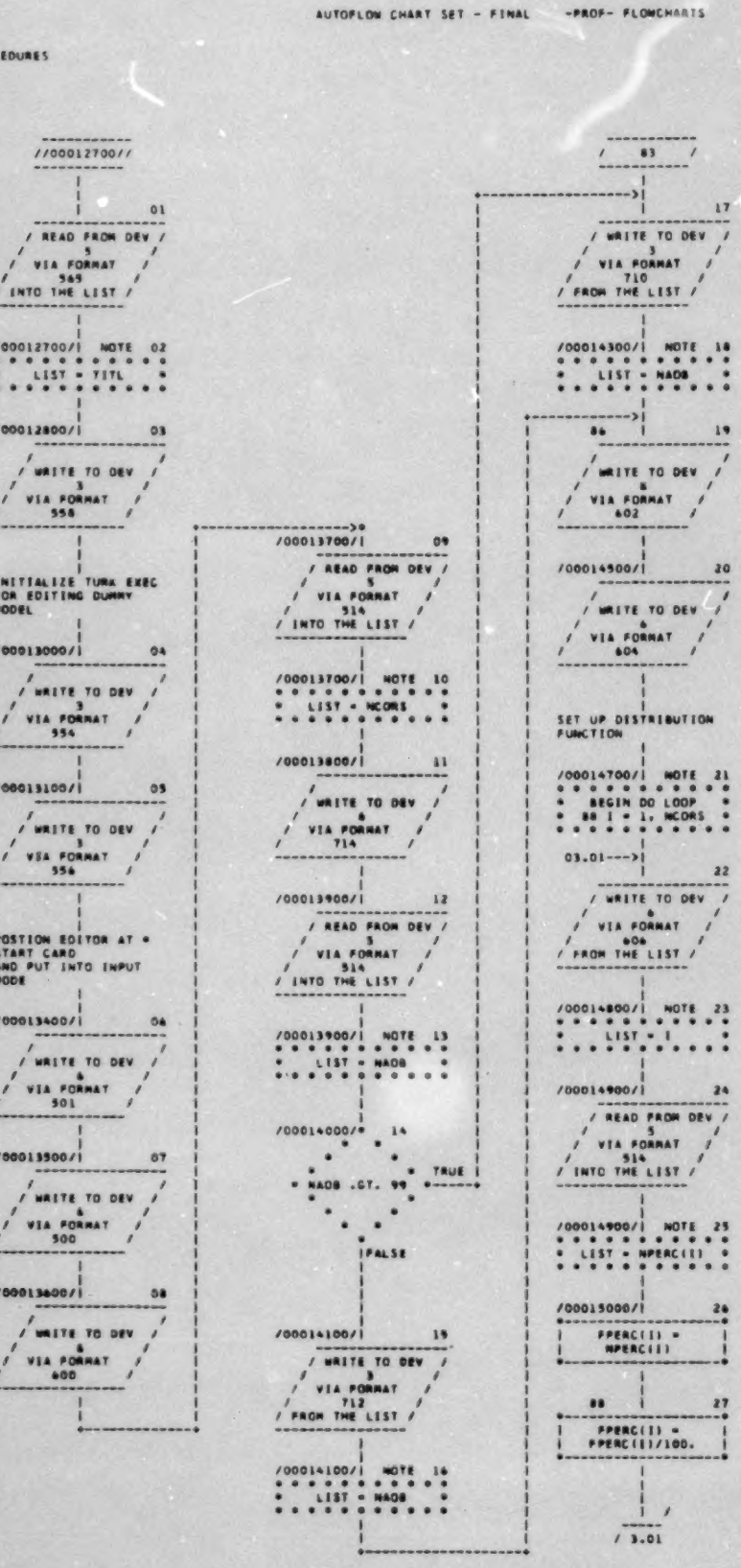


FIGURE III-5

CHART TITLE - PROCEDURES

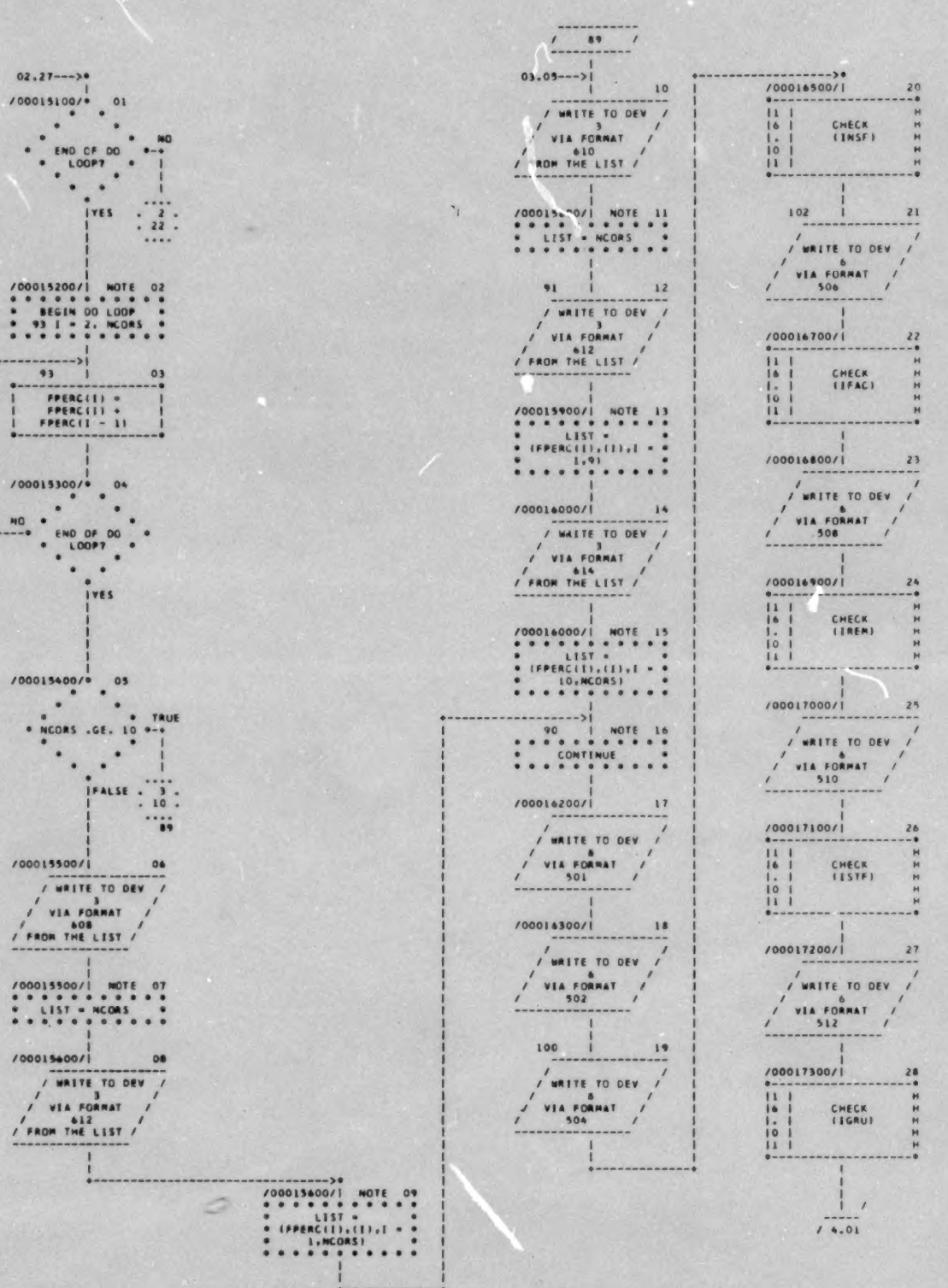


FIGURE III-5 (CONTINUED)

CHART TITLE - PROCEDURES

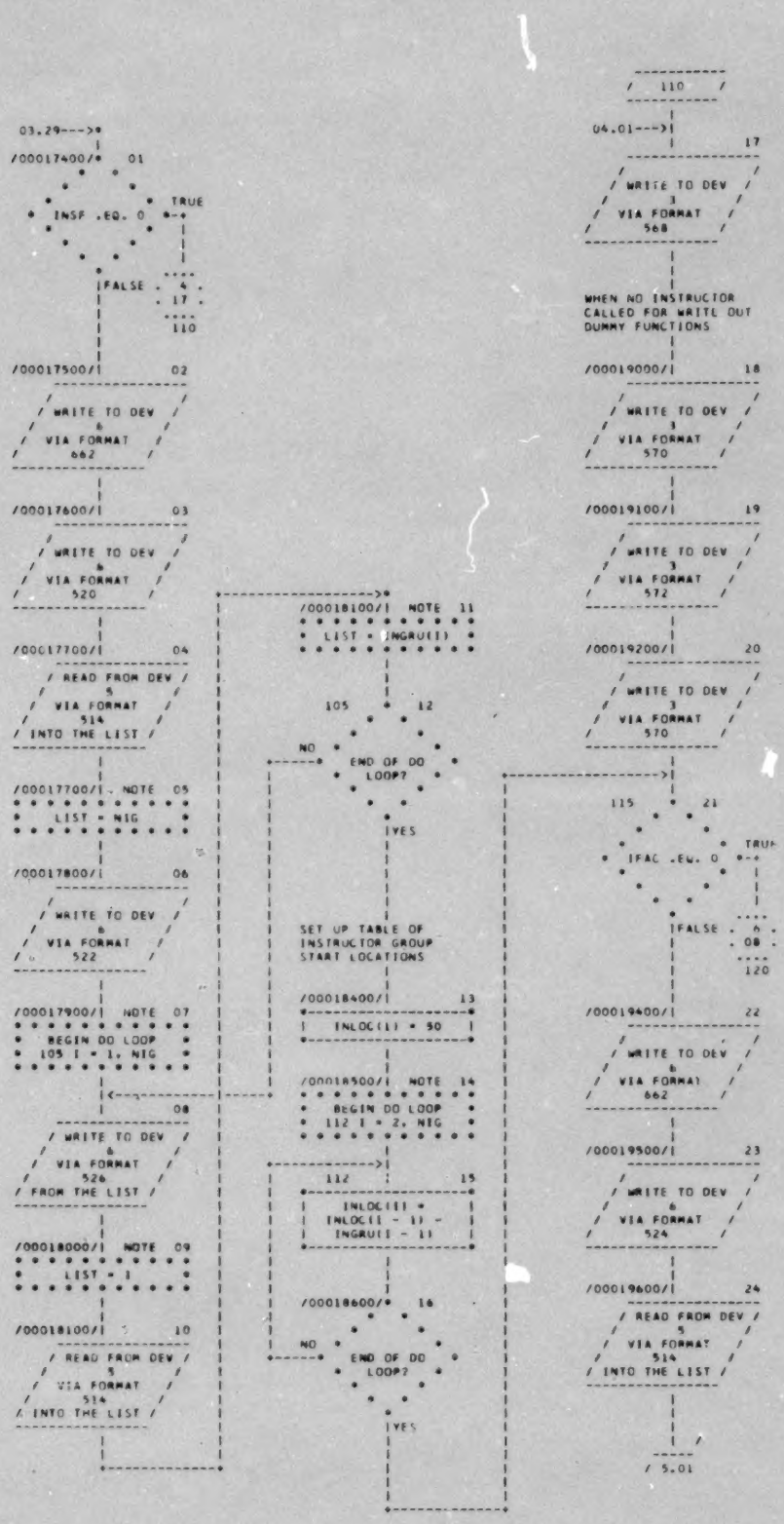
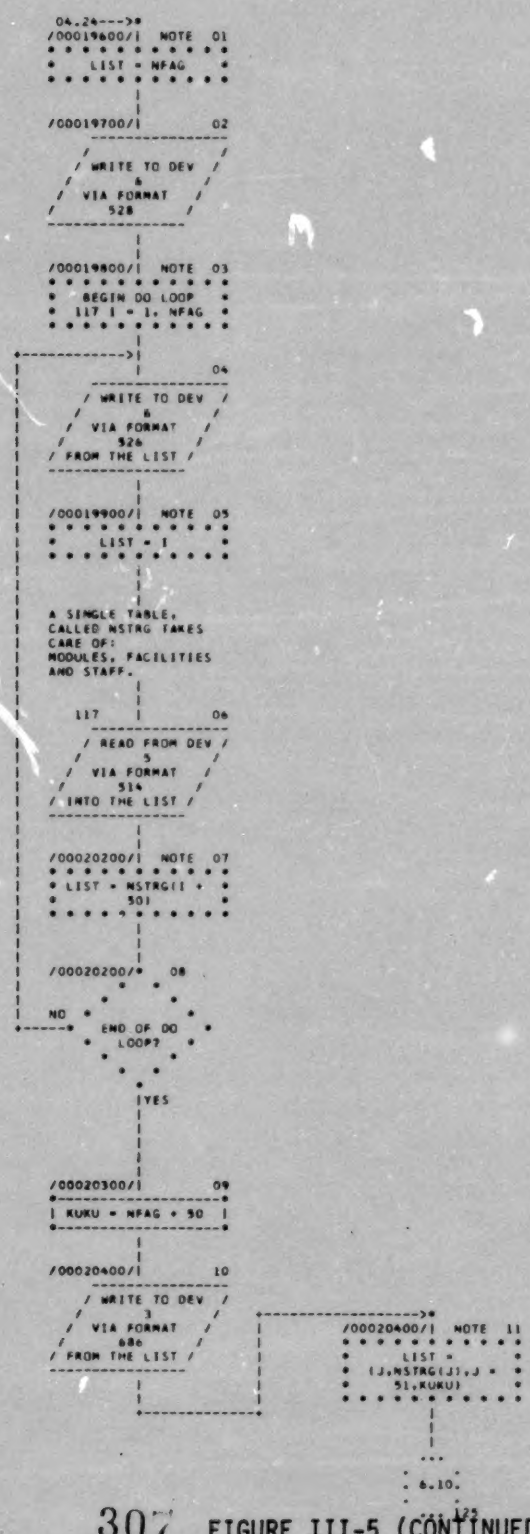


FIGURE III-5 (CONTINUED)

CHART TITLE - PROCEDURES



307 FIGURE III-5 (CONTINUED)

CHART TITLE - PROCEDURES

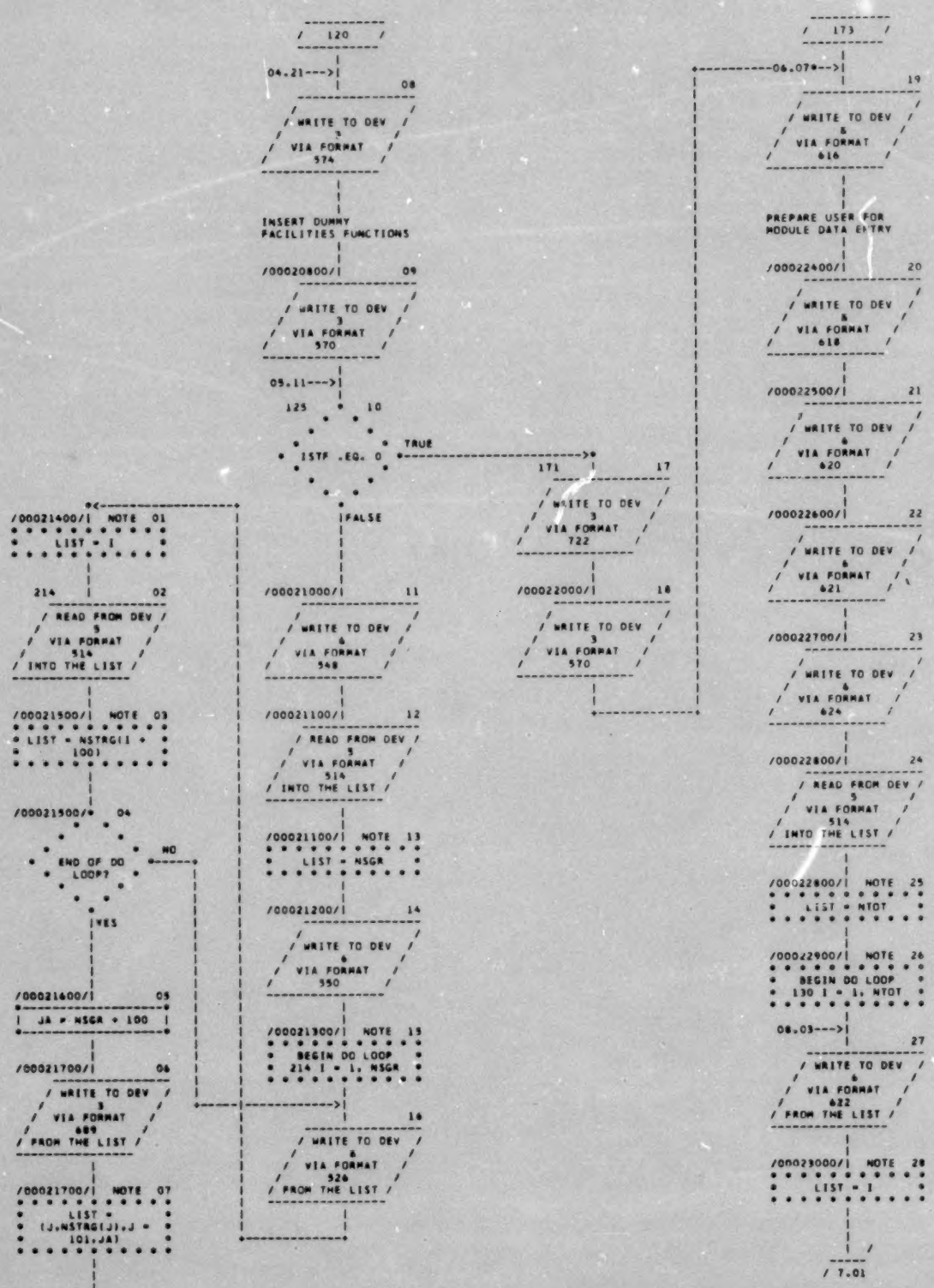


FIGURE III-5 (CONTINUED)

CHART TITLE - PROCEDURES

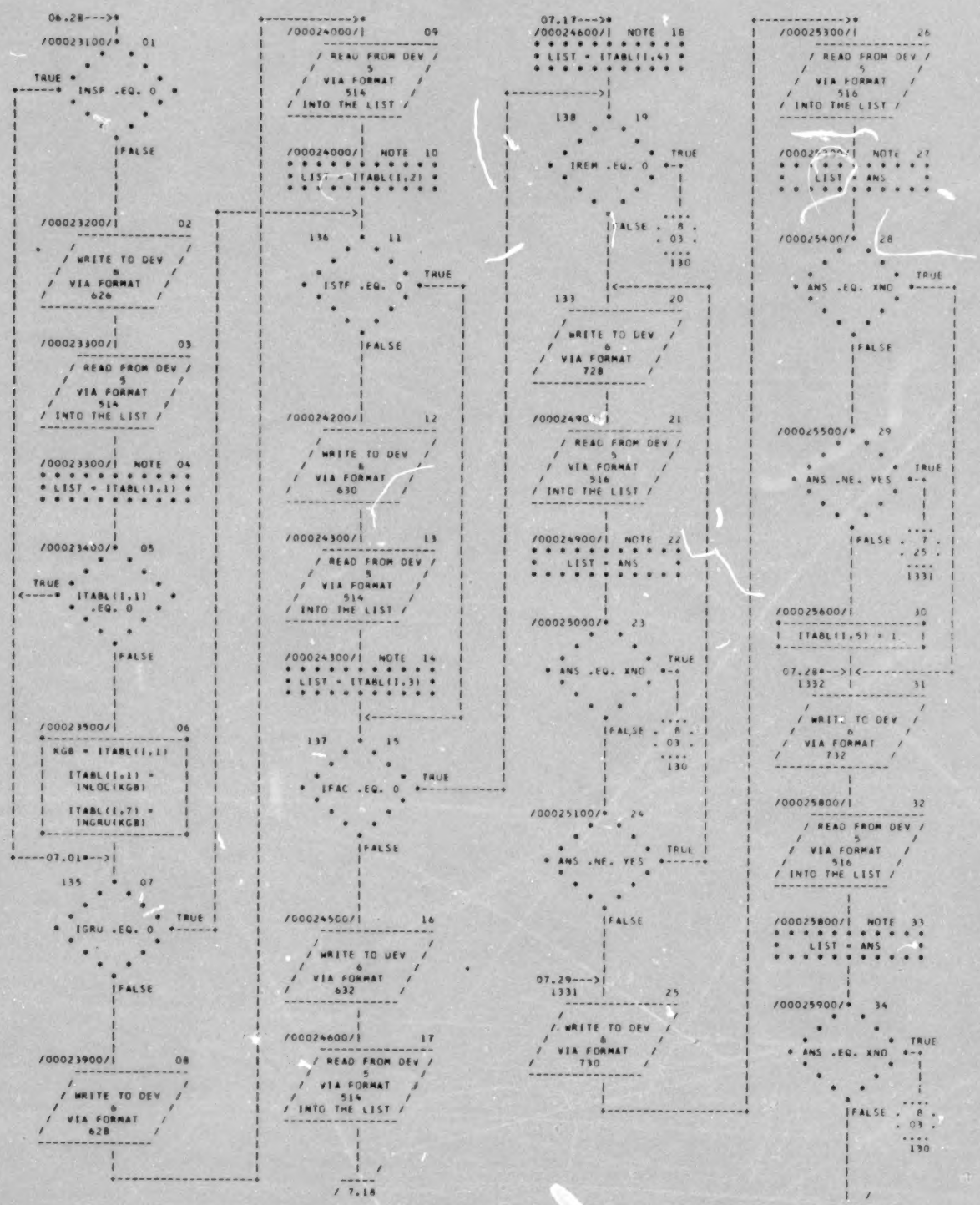


FIGURE III-5 (CONTINUED)

CHART TITLE - PROCEDURES

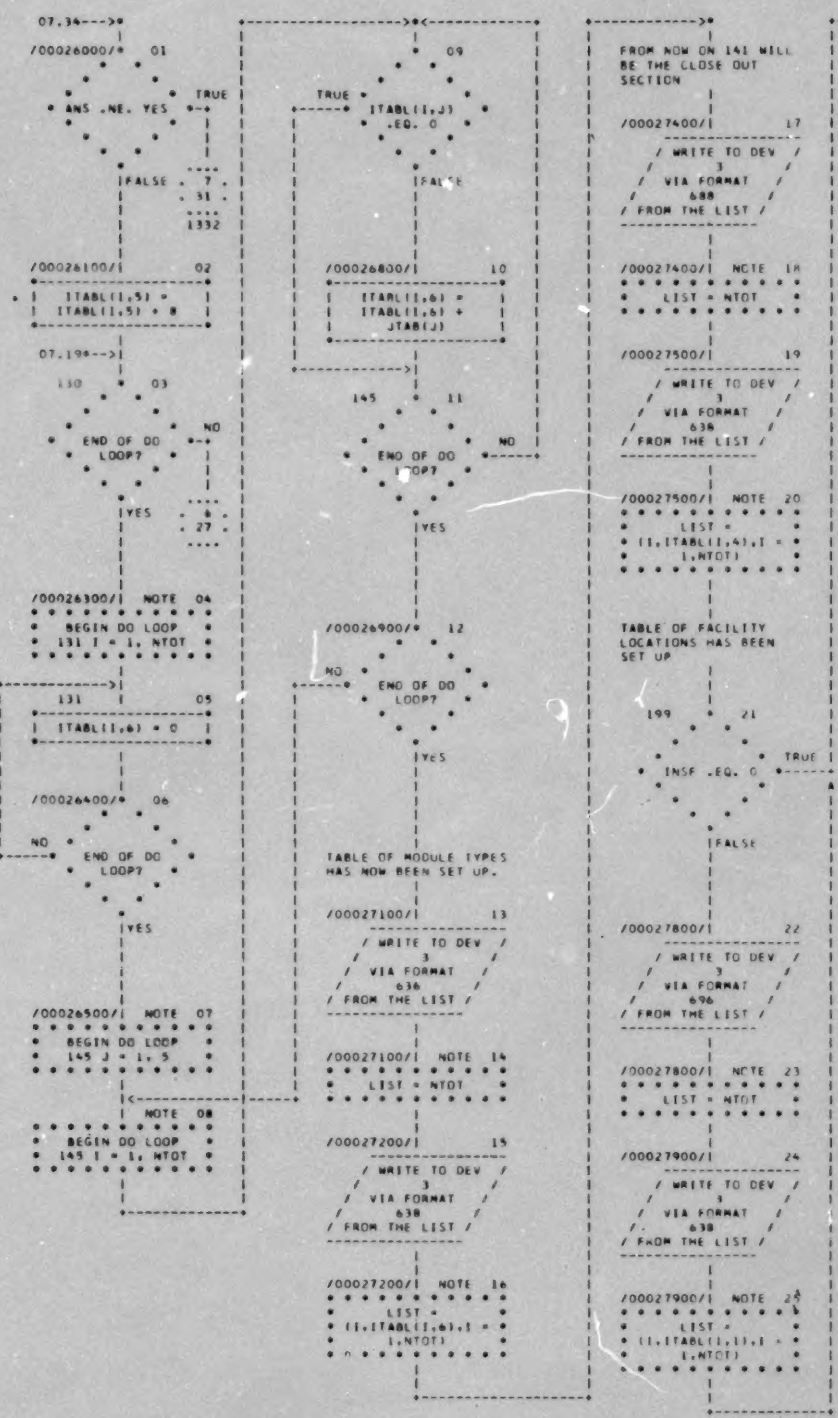


FIGURE III-5 (CONTINUED)

CHART TITLE - PROCEDURES

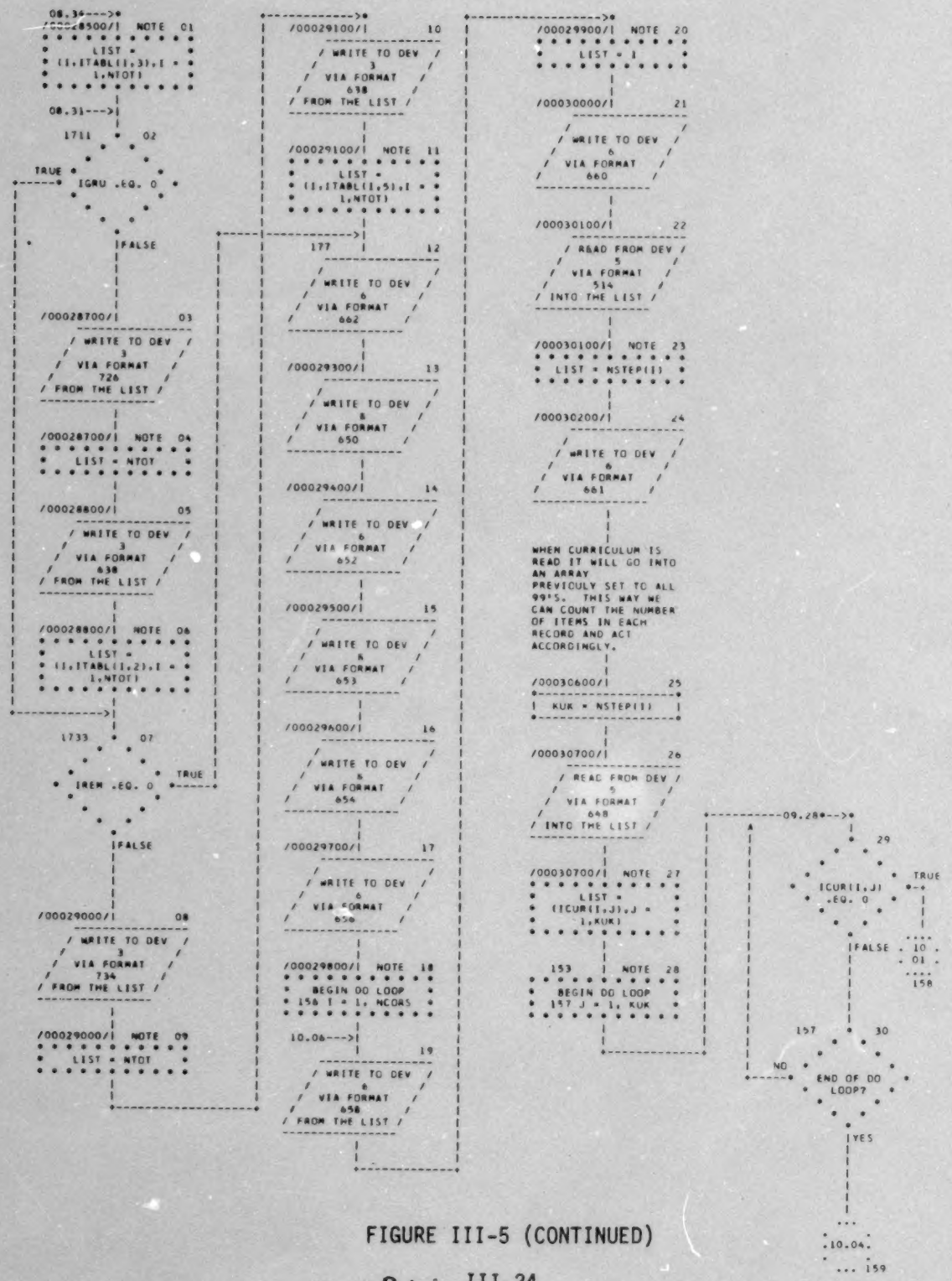


FIGURE III-5 (CONTINUED)

CHART TITLE - PROCEDURES

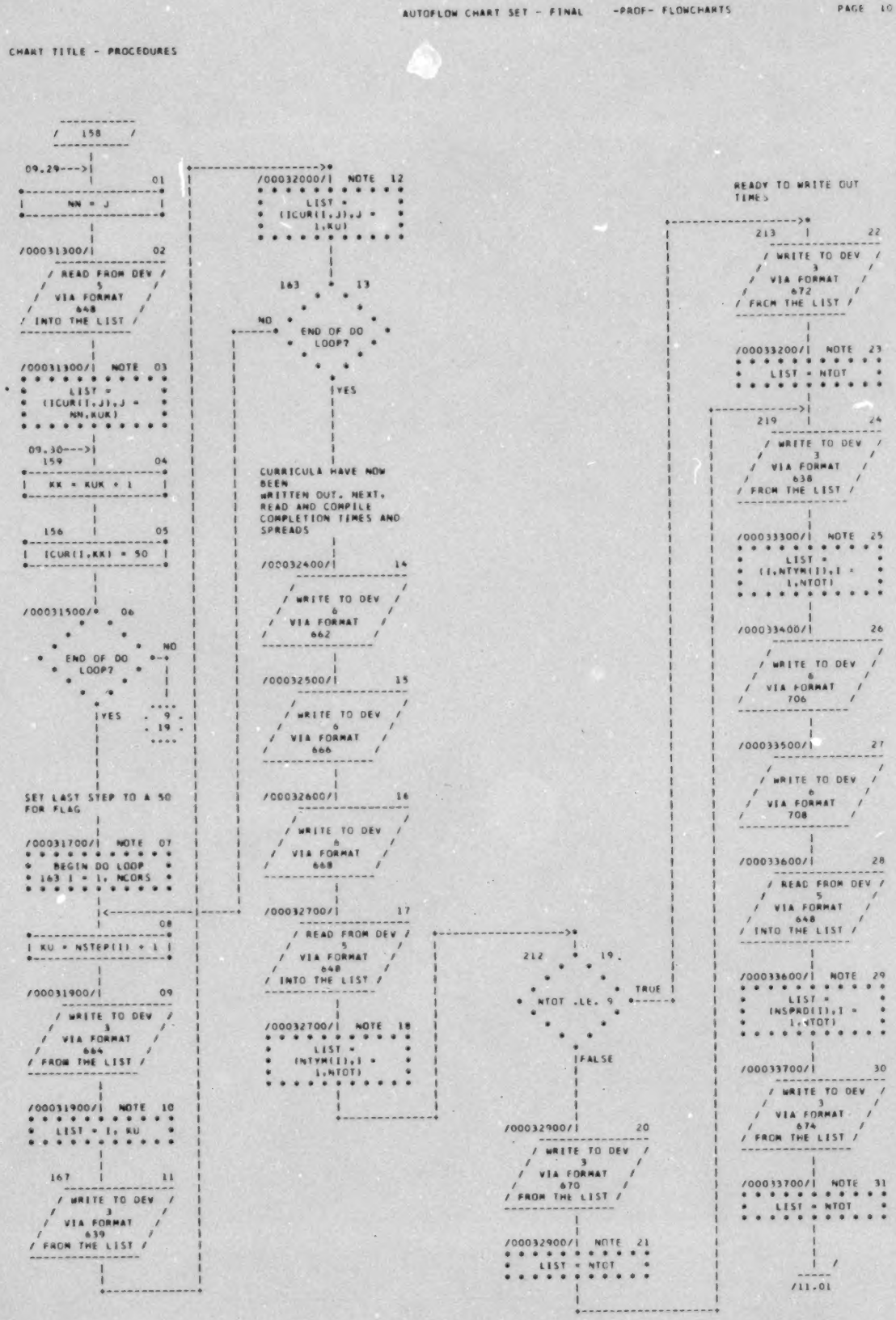


FIGURE III-5 (CONTINUED)

CHART TITLE - PROCEDURES

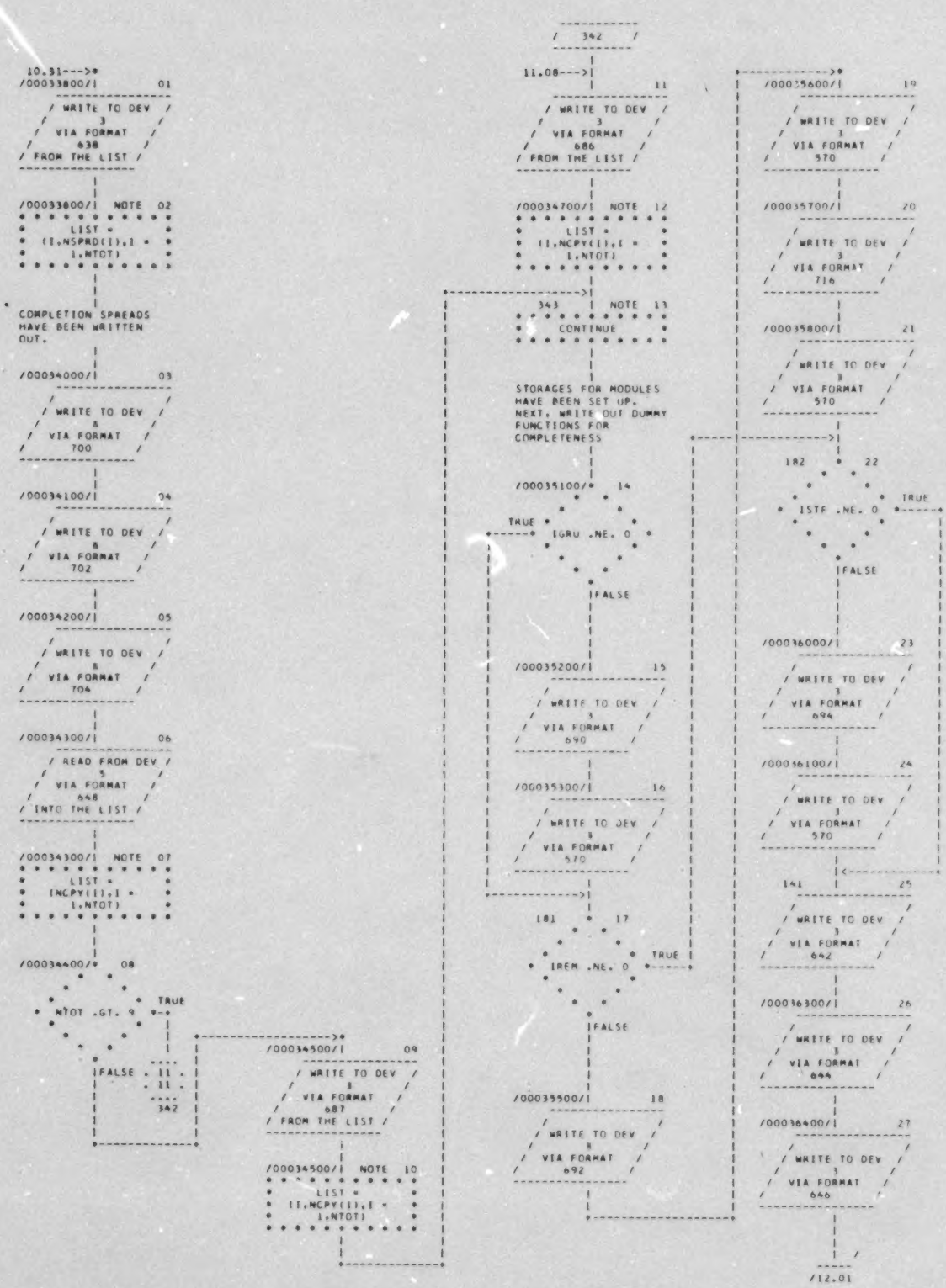


FIGURE III-5 (CONTINUED)

CHART TITLE - PROCEDURES

```
 11.27--->*
/00036500/|        01
     ----------------
    /               /
   / WRITE TO DEV  /
  /       3       /
 /  VIA FORMAT   /
/      644      /
----------------
          |
          |
/00036600/|        02
     ----------------
    /               /
   / WRITE TO DEV  /
  /       3       /
 /  VIA FORMAT   /
/      647      /
----------------
          |
          |
/00036700/|        03
     ----------------
    /               /
   / WRITE TO DEV  /
  /       3       /
 /  VIA FORMAT   /
/      644      /
----------------
          |
          |
/00036800/|        04
     ----------------
    /               /
   / WRITE TO DEV  /
  /       3       /
 /  VIA FORMAT   /
/      718      /
----------------
          |
          |
/00036900/|        05
     ----------------
    /               /
   / WRITE TO DEV  /
  /       3       /
 /  VIA FORMAT   /
/      644      /
----------------
          |
          |
/00037000/|        06
     ----------------
    /               /
   / WRITE TO DEV  /
  /       3       /
 /  VIA FORMAT   /
/      724      /
----------------
          |
          |
/00037100/|        07
     ----------------
    / WRITE TO DEV  /
   /       3       /
  /  VIA FORMAT   /
 /      562      /
/ FROM THE LIST /
----------------
          |
          |
/00037100/|  NOTE  08
* * * * * * * * * * *
*       LIST =      *
* (TITL(I),I = 1,3) *
* * * * * * * * * * *
          |
          |
/00037200/|        09
     ----------------
    /               /
   / WRITE TO DEV  /
  /       3       /
 /  VIA FORMAT   /
/      564      /
----------------
          |
          |
          +------------------------------->*
                                  /00037300/|        10
                                      ----------------
                                     /               /
                                    / WRITE TO DEV  /
                                   /       3       /
                                  /  VIA FORMAT   /
                                 /      566      /
                                 ----------------
                                          |
                                          |
                                 /00037400/| 11
                                      ---------
                                     *  EXIT   *
                                      ---------

                                      CALL EXIT
```


FIGURE III-5 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
500     FORMAT (' COURSE PROFILE DATA')
501     FORMAT(1X)
502     FORMAT ('PLEASE ANSWER THE FOLLOWING QUESTIONS: YES OR NO')
504     FORMAT ('WILL DATA BE NEEDED ON INSTRUCTOR UTILIZATION?')
506     FORMAT ('WILL DATA BE NEEDED ON FACILITIES UTILIZATION?')
508     FORMAT ('ARE SPECIAL REMEDIAL MODULES INCLUDED IN ANY COURSE?')
510     FORMAT ('DO ANY COURSES REQUIRE STAFF MEMBERS (NOT INSTRUCTORS)?')
512     FORMAT ('ARE ANY MODULES GROUP MODULES?')
514     FORMAT (I99)
516     FORMAT (A4)
518     FORMAT (5I2)
520     FORMAT ('HOW MANY INSTRUCTOR GROUPS ARE THERE?')
522     FORMAT ('HOW MANY INSTRUCTORS IN EACH GROUP?')
524     FORMAT ('HOW MANY FACILITIES GROUPS ARE THERE?')
526     FORMAT ('GROUP',I2,'?')
528     FORMAT ('HOW MANY UNITS IN EACH FACILITY GROUP?')
530     FORMAT ('MODULE DATA. ANSWER EACH QUESTION.')
532     FORMAT ('WHEN FINISHED, ENTER WORD STOP')
534     FORMAT ('INSTRUCTOR REQUIRED?')
536     FORMAT ('ENTER INSTRUCTOR GROUP NUMBER.')
538     FORMAT ('FACILITIES REQUIRED?')
540     FORMAT ('ENTER FACILITY NUMBER.')
542     FORMAT ('IS THIS A TEAM MODULE?')
544     FORMAT ('ENTER NUMBER OF TEAM MEMBERS.')
546     FORMAT ('IS STAFF REQUIRED?')
548     FORMAT ('HOW MANY STAFF GROUPS ARE THERE?')
550     FORMAT ('HOW MANY STAFF PERSONNEL IN EACH GROUP?')
552     FORMAT (10(I1,',',I1,'/'))
554     FORMAT ('L /* START')
556     FORMAT ('INPUT')
558     FORMAT ('BBEGSTACK')
560     FORMAT ('PLEASE ENTER THE NAME OF YOUR NEW MODEL')
562     FORMAT ('FILE ',3A4,'GPSS')
564     FORMAT ('BENDSTACK')
565     FORMAT (3A4)
566     FORMAT ('EDIT DUMMY GPSS')
568     FORMAT (' INSTT FUNCTION   P2,D3')
570     FORMAT ('1,2/2,3/3,4')
572     FORMAT (' INSTN FUNCTION   P2,D3')
574     FORMAT (' FACT  FUNCTION   P2,D3')
600     FORMAT ('HOW MANY COURSES ARE THERE?')
602     FORMAT ('WHAT PERCENT OF STUDENTS GO INTO EACH COURSE?')
604     FORMAT ('(ENTER PERCENTAGE AS WHOLE NUMBER)')
606     FORMAT ('COURSE',I2,'?')
608     FORMAT (' STUDT FUNCTION   RN2,D',I1)
610     FORMAT (' STUDT FUNCTION   RN2,D',I2)
612     FORMAT (9(F4.2,',',I1,'/'))
614     FORMAT (9(F4.2,',',I2,'/'))
```

FIGURE III-5 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
616     FORMAT ('1THE FOLLOWING DATA DESCRIBES THE INDIVIDUAL')
618     FORMAT ('COURSE MODULES')
620     FORMAT ('ALL ENTRIES SHOULD BE WHOLE NUMBERS')
621     FORMAT ('1F AN INPUT IS NOT NEEDED, ENTER A ZERO')
622     FORMAT ('MODULE NO. ',I2)
624     FORMAT ('ENTER TOTAL NUMBER OF MODULES')
626     FORMAT ('ENTER INSTRUCTOR GROUP NUMBER')
628     FORMAT ('1F GROUP MODULE, ENTER GROUP SIZE')
630     FORMAT ('1F STAFF REQUIRED, ENTER STAFF GROUP NUMBER')
632     FORMAT ('ENTER FACILITY GROUP NUMBER')
634     FORMAT ('ENTER NUMBER OF ASSOCIATED REMEDIAL MODULE')
636     FORMAT (' MODT  FUNCTION   P2,L',I2)
638     FORMAT (9(I1,',',I2,'/')/(10(I2,',',I2,'/')/))
639     FORMAT (9(',',I2,'/'))
640     FORMAT (10(I2,',',I2,'/'))
642     FORMAT (' ')
644     FORMAT ('U /* START/')
646     FORMAT ('C /. /,/ 50 *')
647     FORMAT ('C /,L /,L/ 50')
648     FORMAT (27I99)
650     FORMAT ('NEXT, CURRICULUM DATA IS TO BE ENTERED')
652     FORMAT ('FIRST, THE NUMBER OF STEPS IN THE SPECIFIED CURRICULUM')
653     FORMAT ('IS ENTERED, THEN MODULE NUMBERS SEPARATED BY COMMAS')
654     FORMAT ('FOR EXAMPLE: 1,3,4,6,7,10 ETC. UNTIL THE COMPLETE')
656     FORMAT ('CURRICULUM HAS BEEN SPECIFIED')
658     FORMAT ('CURRICULUM NO.',I2)
660     FORMAT ('ENTER NUMBER OF STEPS IN THIS CURRICULUM')
661     FORMAT ('NOW ENTER THE CURRICULUM')
662     FORMAT ('1')
664     FORMAT (1X,I2,4X,'FUNCTION   P9,L',I2)
666     FORMAT ('NEXT, ENTER ESTIMATED COMPLETION TIMES FOR EACH MODULE')
668     FORMAT ('ENTRY FORMAT IS THE SAME AS FOR CURRICULUM DATA')
670     FORMAT (' TYME  FUNCTION   P2,L',I2)
672     FORMAT (' TYME  FUNCTION   P2,L',I1)
674     FORMAT (' SPRD  FUNCTION   P2,L',I2)
676     FORMAT (' SPRD  FUNCTION   P2,L',I1)
678     FORMAT ('AFTER COMPLETION TIMES FOR ALL MODULES HAVE BEEN')
680     FORMAT ('ENTERED, EXPECTED VARIATION RANGE IN COMPLETION')
682     FORMAT ('TIMES ARE ENTERED IN THE SAME WAY. A VARIATION MUST')
684     FORMAT ('BE ENTERED EVEN IF IT IS A 0.')
686     FORMAT (7X,'STORAGE',4X,6('S',I2,',',I2,'/'))
687     FORMAT (7X,'STORAGE',4X,6('S',I1,',',I2,'/'))
688     FORMAT (' FACT  FUNCTION   P2,L',I2)
689     FORMAT (7X,'STORAGE',4X,6('S',I3,',',I2,'/'))
690     FORMAT (' TEAM  FUNCTION   P2,L3')
692     FORMAT (' REMM  FUNCTION   P2,L3')
694     FORMAT (' UNTHL FUNCTION   P2,L3')
696     FORMAT (' INSTT FUNCTION   P2,L',I2)
```

FIGURE III-5 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
698     FORMAT (' INSTN FUNCTION   P2,L',I2)
700     FORMAT (/'NOW ENTER THE NUMBER OF COPIES OF INSTRUCTIONAL')
702     FORMAT ('MATTER AVAILABLE FOR EACH MODULE')
704     FORMAT ('ENTRY FORMAT IS THE SAME AS FOR FACILITIES DATA.')
706     FORMAT (/'NOW ENTER COMPLETION VARIATION TIME FOR EACH MODULE.')
708     FORMAT ('SAME FORMAT AS COMPLETION TIMES.')
710     FORMAT (' AOB   VARIABLE   K',I3)
712     FORMAT (' AOB   VARIABLE   K',I2)
714     FORMAT ('ENTER SCHOOL CAPACITY (AOB)')
716     FORMAT (' MOOR  FUNCTION   P2,L3')
718     FORMAT ('C +/S  ++ 50')
720     FORMAT (' UNSHL FUNCTION   P2,L',I2)
722     FORMAT (' UNSHL FUNCTION   P2,L3')
724     FORMAT ('C +/  ++ 50')
726     FORMAT (' TEAM  FUNCTION   P2,L',I2)
728     FORMAT ('DOES THIS MODULE HAVE AN ASSOCIATED REMEDIAL MODULE? (YES
        /NO)')
730     FORMAT ('DOES THE REMEDIAL MODULE REQUIRE AN INSTRUCTOR? YES/NO')
732     FORMAT ('DOES THE REMEDIAL MODULE REQUIRE FACILITIES? (YES/NO)')
734     FORMAT (' MOOR  FUNCTION   P2,L',I2)
        DIMENSION NSTAG(200),NSPRO(30),NCPY(30)
        DIMENSION INGRU(20),TITL(3),INLOC(20),NFGRU(20),LFGRU(20)
        DIMENSION JTAB(5),NSTEP(30)
        DATA YES/'YES'/,XNO/'NO'/,QUIT/'QUIT'/
        DATA JTAB/1,2,4,8,16/
        DIMENSION NPERC(30),FPERC(30),ITABL(30,7),ICUR(15,30),NTYM(30)
        DATA ITABL/210*0/,ICUR/450*99/
```

FIGURE III-5 (CONTINUED)

CHART TITLE - SUBROUTINE CHECK(ITM)

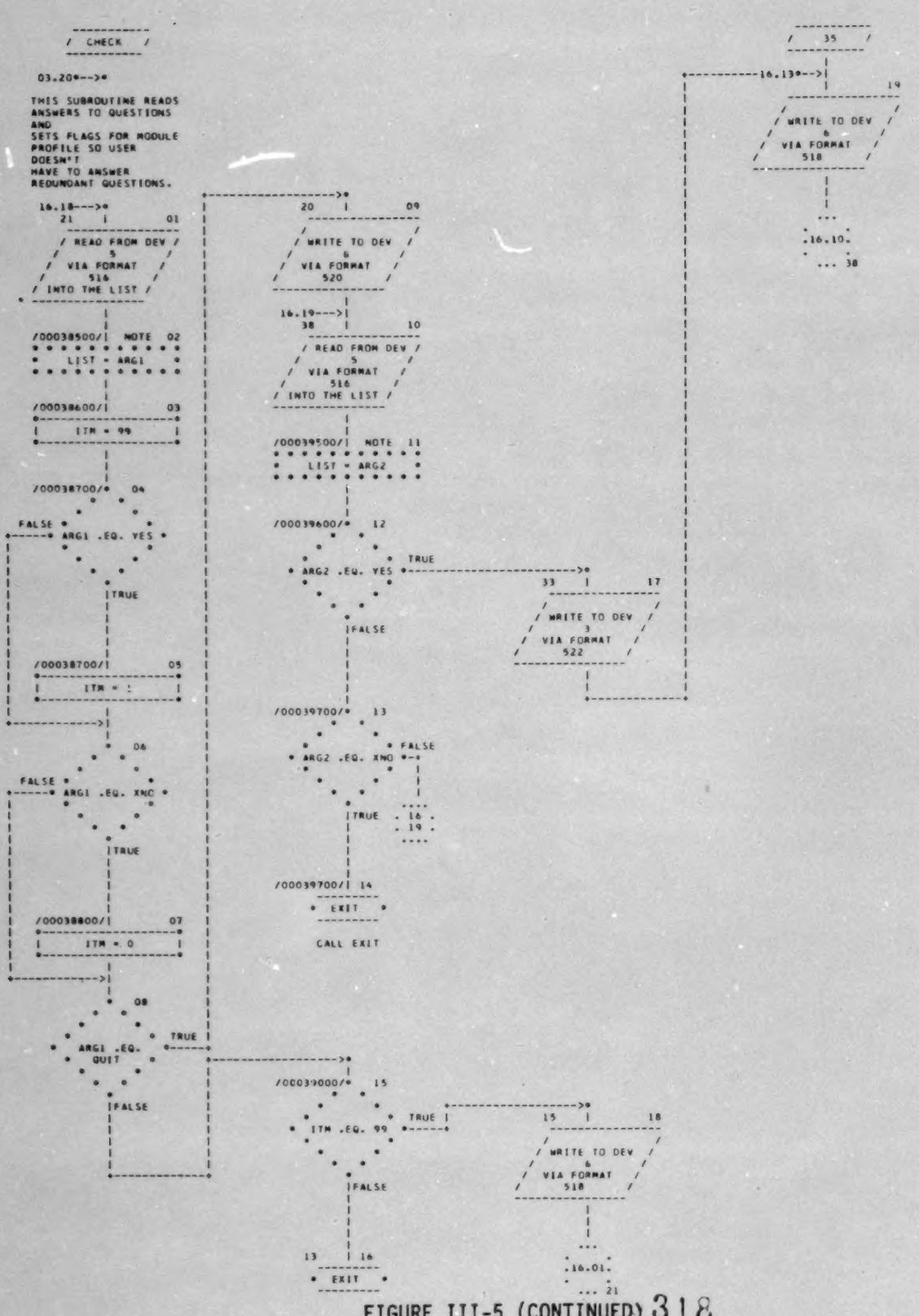


FIGURE III-5 (CONTINUED) 318

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
516     FORMAT (A4)
518     FORMAT ('INPUT NOT UNDERSTOOD, TRY AGAIN.')
522     FORMAT ('FILE')
520     FORMAT('DO YOU WISH TO SAVE THIS CONFIGURATION? ENTER YES OR NO')
        DATA YES/'YES'/,XNO/'NO'/,QUIT/'QUIT'/
```

FIGURE III-5 (CONTINUED)

INPUT LISTING   AUTOFLOW CHART SET - FINAL   -PROF- FLOWCHARTS

FORTRAN MODULE   (LIST,NAMSQ,SOURCSEQ)

CARD NO   ****   CONTENTS   ****

```
  500  FORMAT (' COURSE PROFILE DATA')
  501  FORMAT(1X)
  502  FORMAT ('PLEASE ANSWER THE FOLLOWING QUESTIONS: YES OR NO')
  504  FORMAT ('WILL DATA BE NEEDED ON INSTRUCTOR UTILIZATION?')
  506  FORMAT ('WILL DATA BE NEEDED ON FACILITIES UTILIZATION?')
  508  FORMAT ('ARE SPECIAL REMEDIAL MODULES INCLUDED IN ANY COURSE?')
  510  FORMAT ('DO ANY COURSES REQUIRE STAFF MEMBERS (NOT INSTRUCTORS)?')
  512  FORMAT ('ARE ANY MODULES GROUP MODULES?')
  514  FORMAT (I99)
  516  FORMAT (A4)
  518  FORMAT (5I2)
  520  FORMAT ('HOW MANY INSTRUCTOR GROUPS ARE THERE?')
  522  FORMAT ('HOW MANY INSTRUCTORS IN EACH GROUP?')
  524  FORMAT ('HOW MANY FACILITIES GROUPS ARE THERE?')
  526  FORMAT ('GROUP',I2,'?')
  528  FORMAT ('HOW MANY UNITS IN EACH FACILITY GROUP?')
  530  FORMAT ('MODULE DATA. ANSWER EACH QUESTION.')
  532  FORMAT ('WHEN FINISHED, ENTER WORD STOP')
  534  FORMAT ('INSTRUCTOR REQUIRED?')
  536  FORMAT ('ENTER INSTRUCTOR GROUP NUMBER.')
  538  FORMAT ('FACILITIES REQUIRED?')
  540  FORMAT ('ENTER FACILITY NUMBER.')
  542  FORMAT ('IS THIS A TEAM MODULE?')
  544  FORMAT ('ENTER NUMBER OF TEAM MEMBERS.')
  546  FORMAT ('IS STAFF REQUIRED?')
  548  FORMAT ('HOW MANY STAFF GROUPS ARE THERE?')
  550  FORMAT ('HOW MANY STAFF PERSONNEL IN EACH GROUP?')
  552  FORMAT (10(I1,',',I1,'/'))
  554  FORMAT ('L /* START/')
  556  FORMAT ('INPUT')
  558  FORMAT ('&BEGSTACK')
  560  FORMAT ('PLEASE ENTER THE NAME OF YOUR NEW MODEL')
  562  FORMAT ('FILE ',3A4,'GPSS')
  564  FORMAT ('&ENDSTACK')
  565  FORMAT (3A4)
  566  FORMAT ('EDIT DUMMY GPSS')
  568  FORMAT (' INSTT FUNCTION   P2,D3')
  570  FORMAT ('1,2/2,3/3,4')
  572  FORMAT (' INSTN FUNCTION   P2,D3')
  574  FORMAT (' FACT  FUNCTION   P2,D3')
  600  FORMAT ('HOW MANY COURSES ARE THERE?')
  602  FORMAT ('WHAT PERCENT OF STUDENTS GO INTO EACH COURSE?')
  604  FORMAT ('(ENTER PERCENTAGE AS WHOLE NUMBER)')
  606  FORMAT ('COURSE',I2,'?')
  608  FORMAT (' STUDT FUNCTION   RN2,D',I1)
  610  FORMAT (' STUDT FUNCTION   RN2,D',I2)
  612  FORMAT (9(F4.2,',',I1,'/'))
  614  FORMAT (9(F4.2,',',I2,'/'))
  616  FORMAT ('1THE FOLLOWING DATA DESCRIBES THE INDIVIDUAL')
  618  FORMAT ('COURSE MODULES')
  620  FORMAT ('ALL ENTRIES SHOULD BE WHOLE NUMBERS')
  621  FORMAT ('IF AN INPUT IS NOT NEEDED, ENTER A ZERO')
  622  FORMAT ('MODULE NO. ',I2)
  624  FORMAT ('ENTER TOTAL NUMBER OF MODULES')
  626  FORMAT ('ENTER INSTRUCTOR GROUP NUMBER')
  628  FORMAT ('IF GROUP MODULE, ENTER GROUP SIZE')
  630  FORMAT ('IF STAFF REQUIRED, ENTER STAFF GROUP NUMPER')
  632  FORMAT ('ENTER FACILITY GROUP NUMBER')
  634  FORMAT ('ENTER NUMBER OF ASSOCIATED REMEDIAL MODULE')
  636  FORMAT (' MODT  FUNCTION   P2,L',I2)
  638  FORMAT (9(I1,',',I2,'/')/(10(I2,',',I2,'/')/))
  639  FORMAT (9(',',I2,'/'))
  640  FORMAT (10(I2,',',I2,'/'))
C THESE TWO FORMATS SHOULD HANDLE ALL 'L' TYPE FUNCTIONS
  642  FORMAT (' ')
  644  FORMAT ('U /* START/')
  646  FORMAT ('C /, /,/ 5C *')
  647  FORMAT ('C /,L /,L/ 50')
  648  FORMAT (27I99)
  650  FORMAT ('NEXT, CURRICULUM DATA IS TO BE ENTERED')
  652  FORMAT ('FIRST, THE NUMBER OF STEPS IN THE SPECIFIED CURRICULUM')
  653  FORMAT ('IS ENTERED, THEN MODULE NUMBERS SEPARATED BY COMMAS')
  654  FORMAT ('FOR EXAMPLE: 1,3,4,6,7,10 ETC. UNTIL THE COMPLETE')
  656  FORMAT ('CURRICULUM HAS BEEN SPECIFIED')
  658  FORMAT ('CURRICULUM NO.',I2)
  660  FORMAT ('ENTER NUMBER OF STEPS IN THIS CURRICULUM')
  661  FORMAT ('NOW ENTER THE CURRICULUM')
  662  FORMAT ('1')
  664  FORMAT (1X,I2,4X,'FUNCTION   P9,L',I2)
  666  FORMAT ('NEXT, ENTER ESTIMATED COMPLETION TIMES [illegible] EACH MODULE')
  668  FORMAT ('ENTRY FORMAT IS THE SAME AS FOR CURRICULUM DATA')
  670  FORMAT (' TYME  FUNCTION   P2,L',I2)
  672  FORMAT (' TYME  FUNCTION   P2,L',I1)
  674  FORMAT (' SPRD  FUNCTION   P2,L',I2)
  676  FORMAT (' SPRD  FUNCTION   P2,L',I1)
  678  FORMAT ('AFTER COMPLETION TIMES FOR ALL MODULES HAVE BEEN')
  680  FORMAT ('ENTERED, EXPECTED VARIATION RANGE IN COMPLETION')
  682  FORMAT ('TIMES ARE ENTERED IN THE SAME WAY. A VARIATION MUST')
  684  FORMAT ('BE ENTERED EVEN IF IT IS A 0.')
  686  FORMAT (7X,'STORAGE',4X,6('S',I2,',',I2,'/'))
  687  FORMAT (7X,'STORAGE',4X,6('S',I1,',',I2,'/'))
  688  FORMAT (' FACT  FUNCTION   P2,L',I2)
  689  FORMAT (7X,'STORAGE',4X,6('S',I3,',',I2,'/'))
  690  FORMAT (' TEAM  FUNCTION   P2,L3')
  692  FORMAT (' REMM  FUNCTION   P2,L3')
  694  FORMAT (' UNSHL FUNCTION   P2,L3')
  696  FORMAT (' INSTT FUNCTION   P2,L',I2)
  698  FORMAT (' INSTN FUNCTION   P2,L',I2)
  700  FORMAT (/'NOW ENTER THE NUMBER OF COPIES OF INSTRUCTIONAL')
  702  FORMAT ('MATTER AVAILABLE FOR EACH MODULE')
  704  FORMAT ('ENTRY FORMAT IS THE SAME AS FOR FACILITIES DATA.')
  706  FORMAT (/'NOW ENTER COMPLETION VARIATION TIME FOR EACH MODULE.')
  708  FORMAT ('SAME FORMAT AS COMPLETION TIMES.')
  710  FORMAT (' AOB   VARIABLE   K',I3)
  712  FORMAT (' AOB   VARIABLE   K',I2)
  714  FORMAT ('ENTER SCHOOL CAPACITY (AOB)')
  716  FORMAT (' MODR  FUNCTION   P2,L3')
  718  FORMAT ('C +/S  ++ 50')
  720  FORMAT (' UNSHL FUNCTION   P2,L',I2)
```

TABLE III-3

INPUT LISTING AUTOFLOW CHART SET - FINAL -PROF- FLOWCHARTS

CARD NO **** CONTENTS ****

```
  722  FORMAT (' UNSHL FUNCTION   P2,L3')
  724  FORMAT ('C +/   ++ 50')
  726  FORMAT (' TEAM  FUNCTION   P2,L',I2)
  728  FORMAT ('DOES THIS MODULE HAVE AN ASSOCIATED REMEDIAL MODULE? (YES
      1/NO)')
  730  FORMAT ('DOES THE REMEDIAL MODULE REQUIRE AN INSTRUCTOR? YES/NO')
  732  FORMAT ('DOES THE REMEDIAL MODULE REQUIRE FACILITIES? (YES/NO)')
  734  FORMAT (' MODR  FUNCTION   P2,L',I2)
       DIMENSION NSTRG(200),NSPRO(30),NCPY(30)
       DIMENSION INGRU(20),TTTL(3),INLOC(20),NFGRU(20),LFGRU(20)
       DIMENSION JTAB(5),NSTEP(30)
       DATA YES/'YES'/,XNO/'NO'/,QUIT/'QUIT'/
       DATA JTAB/1,2,4,8,16/
       DIMENSION NPERC(30),FPERC(30),ITABL(30,7),ICUR(15,30),NTYM(30)
       DATA ITABL/210*0/,ICUR/450*99/
C THIS PROGRAM CALLED PROF READS ILS CONFIGURATION DATA IN TUTORIAL
C MODE.  IT SETS UP APPROPRIATE FUNCTIONS AND INSERTS THEM IN ETE MODEL.
       READ (5,565)TITL
       WRITE (3,558)
C INITIALIZE TURK EXEC FOR EDITING DUMMY MODEL
       WRITE (3,554)
       WRITE (3,556)
C POSTION EDITOR AT * START CARD
C AND PUT INTO INPUT MODE
       WRITE (6,501)
       WRITE (6,500)
       WRITE (6,600)
       READ  (5,514)NCORS
       WRITE (6,714)
       READ  (5,514)NAOB
       IF(NAOB.GT.99) GOTO 83
       WRITE (3,712)NAOB
       GOTO 86
 83    WRITE (3,710)NAOB
 86    WRITE (6,602)
       WRITE (6,604)
C SET UP DISTRIBUTION FUNCTION
       DO 88 I = 1,NCORS
       WRITE (6,606)I
       READ (5,514)NPERC(I)
       FPERC(I) = NPERC(I)
 88    FPERC(I) = FPERC(I)/100.
       DO 93 I = 2,NCORS
  93   FPERC(I) = FPERC(I) + FPERC(I-1)
       IF (NCORS.GE.10) GOTO 89
       WRITE (3,608)NCORS
       WRITE (3,612)(FPERC(I),(I),I=1,NCORS)
       GOTO 90
 89    WRITE (3,610)NCORS
 91    WRITE (3,612)(FPERC(I),(I),I=1,9)
       WRITE (3,614)(FPERC(I),(I),I=10,NCORS)
 90    CONTINUE
       WRITE(6,501)
       WRITE (6,502)
 100   WRITE (6,504)
       CALL CHECK(INSF)
 102   WRITE (6,506)
       CALL CHECK (IFAC)
       WRITE (6,508)
       CALL CHECK (IREM)
       WRITE (6,510)
       CALL CHECK (ISTF)
       WRITE (6,512)
       CALL CHECK (IGRU)
       IF (INSF.EQ.0) GOTO 110
       WRITE (6,662)
       WRITE (6,520)
       READ (5,514)NIG
       WRITE (6,522)
       DO 105 I = 1,NIG
       WRITE (6,526)I
       READ (5,514)INGRU (I)
 105   CONTINUE
C SET UP TABLE OF INSTRUCTOR GROUP START LOCATIONS
       INLOC (1) = 50
       DO 112 I = 2,NIG
 112   INLOC (I) = INLOC (I-1)-INGRU (I-1)
       GOTO 115
 110   WRITE (3,568)
C WHEN NO INSTRUCTOR CALLED FOR WRITE OUT DUMMY FUNCTIONS
       WRITE (3,570)
       WRITE (3,572)
       WRITE (3,570)
 115   IF (IFAC.EQ.0)GOTO 120
       WRITE (6,662)
       WRITE (6,524)
       READ (5,514)NFAG
       WRITE (6,528)
       DO 117 I = 1,NFAG
       WRITE (6,526)I
C A SINGLE TABLE, CALLED NSTRG TAKES CARE OF:
C  MODULES, FACILITIES AND STAFF.
 117   READ (5,514)NSTRG(I+50)
       KUKU = NFAG + 50
       WRITE (3,686)(J,NSTRG(J),J = 51,KUKU)
       GOTO 125
 120   WRITE (3,574)
C INSERT DUMMY FACILITIES FUNCTIONS
       WRITE (3,570)
 125   IF (ISTF.EQ.0) GOTO 171
       WRITE (6,548)
       READ  (5,514)NSGR
       WRITE (6,550)
       DO 214 I = 1,NSGR
       WRITE (6,526)I
 214   READ  (5,514)NSTRG(I+100)
       JA = NSGR + 100
       WRITE (3,689)(J,NSTRG(J), J = 101,JA)
       GOTO 173
 171   WRITE (3,722)
       WRITE (3,570)
       GOTO 173
```


TABLE III-3 (CONTINUED)

INPUT LISTING AUTOFLOW CHART SET - FINAL -PROF- FLOWCHARTS

```
CARD NO    ****                        CONTENTS                         ****

        173  WRITE (6,616)                                              00022200
       C PREPARE USER FOR MODULE DATA ENTRY                             00022300
             WRITE (6,618)                                              00022400
             WRITE (6,620)                                              00022500
             WRITE (6,621)                                              00022600
             WRITE (6,624)                                              00022700
             READ  (5,514)NTOT                                          00022800
             DO 130 I = 1,NTOT                                          00022900
             WRITE (6,622)I                                             00023000
             IF (INSF.EQ.0) GOTO 135                                    00023100
             WRITE (6,626)                                              00023200
             READ  (5,514)ITABL(I,1)                                    00023300
             IF (ITABL(I,1).EQ.0) GOTO 135                              00023400
             KGB = ITABL(I,1)                                           00023500
             ITABL (I,1) = INLOC(KGB)                                   00023600
             ITABL(I,7) = INGRU (KGB)                                   00023700
        135  IF(IGRU.EQ.0) GOTO 136                                     00023800
             WRITE (6,628)                                              00023900
             READ  (5,514)ITABL(I,2)                                    00024000
        136  IF (ISTF.EQ.0)GOTO 137                                     00024100
             WRITE (6,630)                                              00024200
             READ  (5,514)ITABL(I,3)                                    00024300
        137  IF(IFAC.EQ.0) GOTO 138                                     00024400
             WRITE (6,632)                                              00024500
             READ  (5,514)ITABL(I,4)                                    00024600
        138  IF(IREM.EQ.0)GOTO 130                                      00024700
        133  WRITE  (6,728)                                             00024800
             READ (5,516)ANS                                            00024900
             IF (ANS.EQ.XNO) GOTO 130                                   00025000
             IF (ANS.NE.YES) GOTO 133                                   00025100
        1331 WRITE (6,730)                                              00025200
             READ (5,516)ANS                                            00025300
             IF (ANS.EQ.XNO) GOTO 1332                                  00025400
             IF(ANS.NE.YES) GOTO 1331                                   00025500
             ITABL(I,5) = 1                                             00025600
        1332 WRITE (6,732)                                              00025700
             READ  (5,516)ANS                                           00025800
             IF (ANS.EQ.XNO) GOTO 130                                   00025900
             IF (ANS.NE.YES) GOTO 1332                                  00026000
             ITABL (I,5) = ITABL (I,5)+8                                00026100
        130  CONTINUE                                                   00026200
             DO 131 I = 1,NTOT                                          00026300
        131  ITABL (I,6) = 0                                            00026400
             DO 145 J = 1,5                                             00026500
             DO 145 I = 1,NTOT                                          00026600
             IF (ITABL(I,J).EQ.0) GOTO 145                              00026700
             ITABL(I,6) = ITABL(I,6) + JTAB(J)                          00026800
        145  CONTINUE                                                   00026900
       C TABLE OF MODULE TYPES HAS NOW BEEN SET UP.                     00027000
             WRITE (3,636)NTOT                                          00027100
             WRITE (3,638)(I,ITABL(I,6),I = 1,NTOT)                     00027200
       C FROM NOW ON 141 WILL BE THE CLOSE OUT SECTION                  00027300
             WRITE (3,688)NTOT                                          00027400
             WRITE (3,638)(I,ITABL(I,4),I = 1,NTOT)                     00027500
       C TABLE OF FACILITY LOCATIONS HAS BEEN SET UP                    00027600
        199  IF(INSF.EQ.0) GOTO 193                                     00027700
             WRITE (3,696)NTOT                                          00027800
             WRITE (3,638)(I,ITABL(I,1),I = 1,NTOT)                     00027900
        193  IF(IFAC.EQ.0) GOTO 194                                     00028000
             WRITE (3,698)NTOT                                          00028100
             WRITE (3,638)(I,ITABL(I,7),I = 1,NTOT)                     00028200
        194  IF(ISTF.EQ.0) GOTO 1711                                    00028300
             WRITE (3,720)NTOT                                          00028400
             WRITE (3,638)(I,ITABL(I,3),I = 1,NTOT)                     00028500
        1711 IF (IGRU.EQ.0) GOTO 1733                                   00028600
             WRITE (3,726) NTOT                                         00028700
             WRITE (3,638)(I,ITABL(I,2),I = 1,NTOT)                     00028800
        1733 IF(IREM.EQ.0) GOTO 17[illegible]                           00028900
             WRITE (3,734)NTOT                                          00029000
             WRITE (3,638)(I,ITABL(I,5),I = 1,NTOT)                     00029100
        177  WRITE (6,662)                                              00029200
             WRITE (6,650)                                              00029300
             WRITE (6,652)                                              00029400
             WRITE (6,653)                                              00029500
             WRITE (6,654)                                              00029600
             WRITE (6,656)                                              00029700
             DO 156 I = 1,NCORS                                         00029800
             WRITE (6,658)I                                             00029900
             WRITE (6,660)                                              00030000
             READ  (5,514)NSTEP (I)                                     00030100
             WRITE (6,661)                                              00030200
       C WHEN CURRICULUM IS READ IT WILL GO INTO AN ARRAY                00030300
       C  PREVIOULY SET TO ALL 99'S.  THIS WAY WE CAN COUNT THE NUMBER   00030400
       C   OF ITEMS IN EACH RECORD AND ACT ACCORDINGLY.                 00030500
             KUK = NSTEP(I)                                             00030600
             READ (5,648)(ICUR(I,J),J = 1,KUK)                          00030700
        153  DO 157 J = 1,KUK                                           00030800
             IF(ICUR(I,J).EQ.0) GOTO 158                                00030900
        157   CONTINUE                                                  00031000
             GOTO 159                                                   00031100
        158  NN = J                                                     00031200
             READ (5,648)(ICUR(I,J),J = NN,KUK)                         00031300
        159  KK = KUK + 1                                               00031400
        156  ICUR(I,KK) = 50                                            00031500
       C SET LAST STEP TO A 50 FOR FLAG                                 00031600
             DO 163 I = 1,NCORS                                         00031700
             KU = NSTEP(I) + 1                                          00031800
             WRITE (3,664)I , KU                                        00031900
        167   WRITE (3,639)(ICUR(I,J),J=1,KU)                           00032000
        163  CONTINUE                                                   00032100
       C CURRICULA HAVE NOW BEEN                                        00032200
       C  WRITTEN OUT. NEXT, READ AND COMPILE COMPLETION TIMES AND SPREADS 00032300
             WRITE (6,662)                                              00032400
             WRITE (6,666)                                              00032500
             WRITE (6,668)                                              00032600
             READ (5,648)(NTYM(I), I=1,NTOT)                            00032700
        212  IF(NTOT.LE.9) GOTO 213                                     00032800
             WRITE (3,670)NTOT                                          00032900
             GOTO 219                                                   00033000
       C READY TO WRITE OUT TIMES                                       00033100
        213  WRITE (3,672)NTOT                                          00033200
        219  WRITE (3,638)(I,NTYM(I),I = 1,NTOT)                        00033300
```


TABLE III-3 (CONTINUED)

INPUT LISTING AUTOFLOW CHART SET - FINAL -PROF- FLOWCHARTS

CARD NO **** CONTENTS ****

```
      WRITE (6,706)                                                    00033400
      WRITE (6,708)                                                    00033500
      READ  (5,648)(NSPRD (I),I = 1,NTOT)                              00033600
      WRITE (3,674)NTOT                                                00033700
      WRITE (3,638)(I,NSPRD(I),I = 1,NTOT)                             00033800
C COMPLETION SPREADS HAVE BEEN WRITTEN OUT.                            00033900
      WRITE (6,700)                                                    00034000
      WRITE (6,702)                                                    00034100
      WRITE (6,704)                                                    00034200
      READ  (5,648)(NCPY(I),I = 1,NTOT)                                00034300
      IF (NTOT.GT.9) GOTO 342                                          00034400
      WRITE (3,687)(I,NCPY(I),I = 1,NTOT)                              00034500
      GOTO 343                                                         00034600
  342 WRITE (3,686)(I,NCPY(I),I = 1,NTOT)                              00034700
  343 CONTINUE                                                         00034800
C STORAGES FOR MODULES HAVE BEEN SET UP.                               00034900
C  NEXT, WRITE OUT DUMMY FUNCTIONS FOR COMPLETENESS                    00035000
      IF(IGRU.NE.0) GOTO 181                                           00035100
      WRITE (3,690)                                                    00035200
      WRITE (3,570)                                                    00035300
  181 IF(IREM.NE.0) GOTO 182                                           00035400
      WRITE (3,692)                                                    00035500
      WRITE (3,570)                                                    00035600
      WRITE (3,716)                                                    00035700
      WRITE (3,570)                                                    00035800
  182 IF(ISTF.NE.0)GOTO 141                                            00035900
      WRITE (3,694)                                                    00036000
      WRITE (3,570)                                                    00036100
  141 WRITE (3,642)                                                    00036200
      WRITE (3,644)                                                    00036300
      WRITE (3,646)                                                    00036400
      WRITE (3,644)                                                    00036500
      WRITE (3,647)                                                    00036600
      WRITE (3,644)                                                    00036700
      WRITE (3,718)                                                    00036800
      WRITE (3,644)                                                    00036900
      WRITE (3,724)                                                    00037000
      WRITE (3,562)(TITL(I),I = 1,3)                                   00037100
      WRITE (3,564)                                                    00037200
      WRITE (3,566)                                                    00037300
      CALL EXIT                                                        00037400
      END                                                              00037500
      SUBROUTINE CHECK (ITM)                                           00037600
C THIS SUBROUTINE READS ANSWERS TO QUESTIONS AND                       00037700
C SETS FLAGS FOR MODULE PROFILE SO USER DOESN'T                        00037800
C HAVE TO ANSWER REDUNDANT QUESTIONS.                                  00037900
  516 FORMAT (A4)                                                      00038000
  518 FORMAT ('INPUT NOT UNDERSTOOD, TRY AGAIN.')                      00038100
  522 FORMAT ('FILE')                                                  00038200
  520 FORMAT('DO YOU WISH TO SAVE THIS CONFIGURATION? ENTER YES OR NO') 00038300
      DATA YES/'YES'/,XNO/'NO'/,QUIT/'QUIT'/                           00038400
   21 READ ( 5,516)ARG1                                                00038500
      ITM = 99                                                         00038600
      IF (ARG1.EQ.YES) ITM = 1                                         00038700
      IF (ARG1.EQ.XNO) ITM = 0                                         00038800
      IF (ARG1.EQ.QUIT) GOTO 20                                        00038900
      IF (ITM.EQ.99) GOTO 15                                           00039000
   13 RETURN                                                           00039100
   15 WRITE (6,518)                                                    00039200
      GOTO 21                                                          00039300
   20 WRITE (6,520)                                                    00039400
   38 READ (5,516)ARG2                                                 00039500
      IF (ARG2.EQ.YES) GOTO 33                                         00039600
      IF (ARG2.EQ.XNO) CALL EXIT                                       00039700
   35 WRITE (6,518)                                                    00039800
      GOTO 38                                                          00039900
   33 WRITE (3,522)                                                    00040000
      GOTO 35                                                          00040100
      END                                                              00040200
```

TABLE III-3 (CONTINUED)

INPUT CARD SPECIFICATIONS

TABLE NAME	
CURRICULUM CARDS	1 FUNCTION P0.L0
	1.1/2.2/3.4/4.5
	2 FUNCTION P0.L0
	1.2/2.4/3.5/4.6/5.7
MODULE TYPE CARDS	MODT FUNCTION P2.L0
	1.0/2.17/3.0/4.4/5.10/6.3/7.4/8.10
COMPLETION TIME CARDS	TYME FUNCTION P2.L0
	1.6/2.0/3.4/4.7/5.0/6.4/7.5/8.3
COMPLETION TIME VARIATION CARDS	SPRD FUNCTION P2.L0
	1.2/2.3/3.1/4.1/5.2/6.1/7.1/8.1
INSTRUCTOR ASSIGNMENT CARDS	INSTT FUNCTION P2.L0
	1.3/2.3/3.3/4.3/5.0/6.0/7.0/8.0
INSTRUCTOR QUALIFICATION CARDS	INSTN FUNCTION P2.L0
	1.2/2.2/3.2/4.2/5.2/6.2/7.2/8.2
MODULE LOCATION CARDS	MODFL FUNCTION P2.L0
	1.30/2.45/3.55/4.70/5.75/6.80/7.115/8.140
MODULE INVENTORY CARDS	MODN FUNCTION P2.L0
	1.10/2.10/3.0/4.10/5.5/6.15/7.20/8.10
REMEDIAL MODULE LOCATION CARDS	MODRL FUNCTION P2.L0
	1.0/2.100/3.0/4.0/5.0/6.0/7.0/8.120
REMEDIAL MODULE INVENTORY CARDS	MODRN FUNCTION P2.L0
	1.0/2.15/3.0/4.0/5.0/6.0/7.0/8.10
REMEDIAL MODULE TYPE CARDS	MODR FUNCTION P2.L0
	1.0/2.1/3.0/4.0/5.0/6.0/7.0/8.1
OTHER FACILITIES ASSIGNMENTS	FACT FUNCTION P2.L0
	1.1/2.0/3.0/4.0/5.2/6.0/7.0/8.0
STUDENT DISTRIBUTION CARDS	STUDT FUNCTION RM2.D3
	.30.1/.75.2/1.0.3
OTHER FACILITIES INVENTORY	STORAGE S1.50/S2.75
TEAM DEFINITION CARDS	TEAM FUNCTION P2.L0
	1.0/2.0/3.0/4.0/5.2/6.4/7.0/8.0

FIGURE III-6

MODEL UPDATE

Types of Update. There are two basic types of updates: when a new source card is generated to replace an existing source card and when an Edit Change command is generated to act upon an existing source statement.

MODEL UPDATE SUBSYSTEM. Once an executable GPSS model has been generated via the Model Generation Subsystem, the user may desire to vary some of the model parameters to obtain simulation runs for different ILS configurations. The Model Update Subsystem provides a means for making changes to an existing model without requiring all the steps needed to generate a model from scratch.

This subsystem consists of a main element: the FORTRAN program called MAD that interrogates the user as to changes required, the control executive (ETE2), a dummy executive (TURK) and the NCSS Edit Program. Figure III-7 is the Autoflow diagram for the update subsystem. MAD uses the model name, supplied by ETE2, to locate the appropriate GPSS source file for update. Initially, the program displays a menu of eight possible changes and branches to the correct program section based on the user response via a computed GOTO statement.

The most complex update activity is curriculum update. This is because module insertion or deletion requires a change in the length of the curriculum itself. Each curriculum is made up of a function designator card which specifies the curriculum number and length, and some number of follower cards (see Figure III-6).

Insertion or deletion requires that the appropriate function designator card be located and the length designation increased or decreased by one. A replacement card is then generated with the correct length designation field.

Insertion, replacement or deletion is done by generating a CSS Edit Change command. Since this command permits specification of the number of lines to be scanned while attempting to make the specified change, it is necessary to count the cards following the function designator card. The number of function follower cards is then inserted in the change command and the command written into the stack of edit commands contained in TURK EXEC. A sample of a set of commands to delete a module from a curriculum is shown in Table III-4.

The edit commands listed in Table III-4 are representative of the procedures used for most update procedures and perform the following steps:

a. Move the edit pointer to the top of the file.
b. Locate (i.e. move the edit pointer) to function number 1. Note that the location procedure assumes the GPSS statements are in fixed format, not free-form as is permitted under GPSS V.
c. Replace the existing function definition statement with a new statement having a length designation field decremented by 1.
d. Move the edit pointer to the next record.
e. Remove the field: /,3 from the record indicated by the edit pointer.
f. Replace the file FIRE GPSS with the updated file and return to CSS control.

Change card structure. MAD and PROF use different techniques for structuring change cards. GPSS does not permit embedded blanks in function follower cards (and the majority of initial inputs and changes are function follower cards) so both programs must structure these cards with no embedded blanks.

MAD avoids embedded blanks by testing all input field sizes (e.g.9,99 etc) and by using an output FORMAT statement structured with no empty output columns.

PROF uses a limited number of different output FORMAT statements and produces function follower cards which may have 1 (but never more than 1) embedded blank per field. These blanks are later removed during the edit phase. PROF writes two sets of edit commands at the end of the stack. The function of these commands is to reposition the edit pointer at the beginning of the input section (at the * START card) and to search for character combinations containing embedded blanks (eg:, 3,) and remove the blank.

Because the length specification change is not incorporated into the model until after editing is finished, only one insertion or deletion can be made in a given curriculum, during a single execution of the update subsystem. For example, if two insertions are attempted in the same curriculum, during a single run the second insertion will be made, and a new length specification calculated, based on the length specification that existed at the start of the run and will produce an erroneous new length specification.

```
TOP
L /   2     FUNC/
R    2     FUNCTION    P9,L7
D 1
C !/,3!!1
FILE ETE      GPSS
```

SAMPLE EDIT COMMANDS

TABLE III-4

CHART TITLE - PROCEDURES

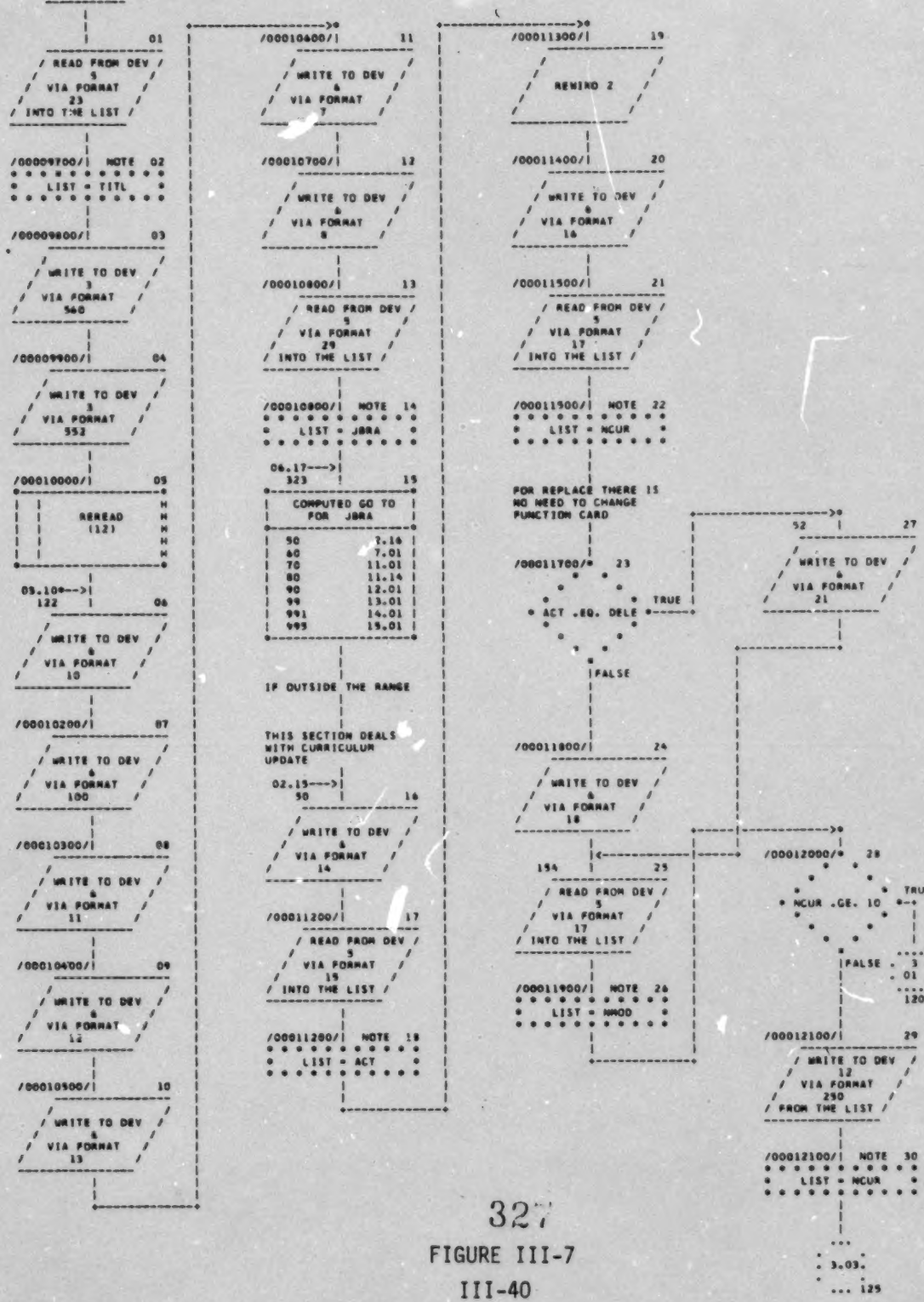


FIGURE III-7

CHART TITLE - PROCEDURES

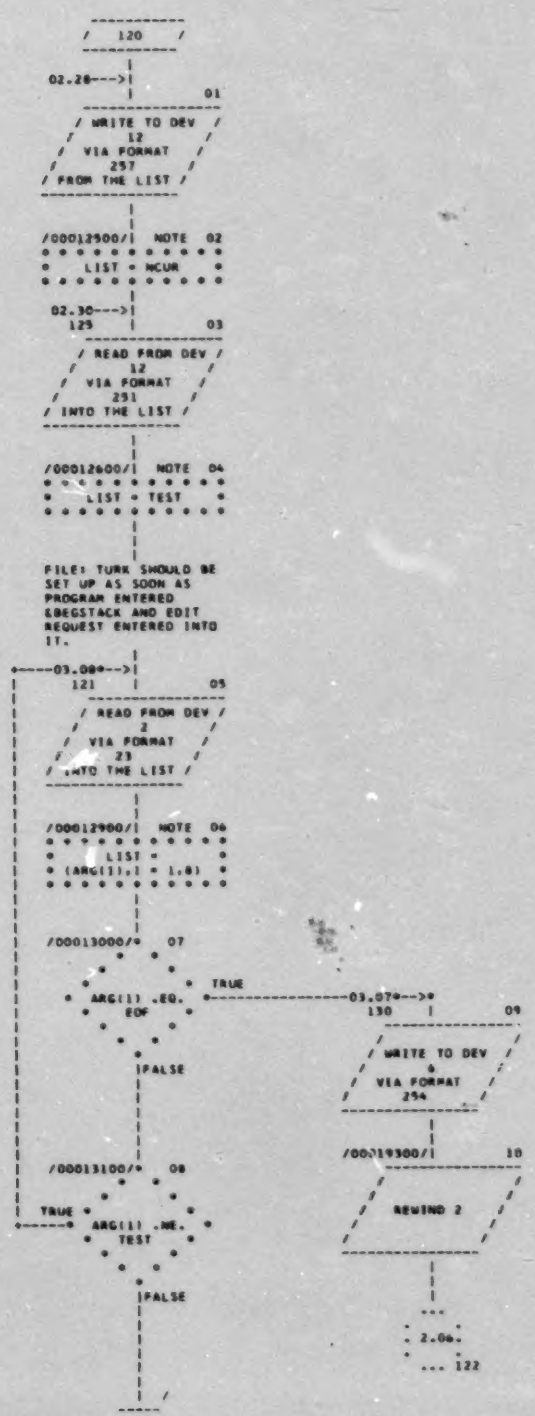


FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES

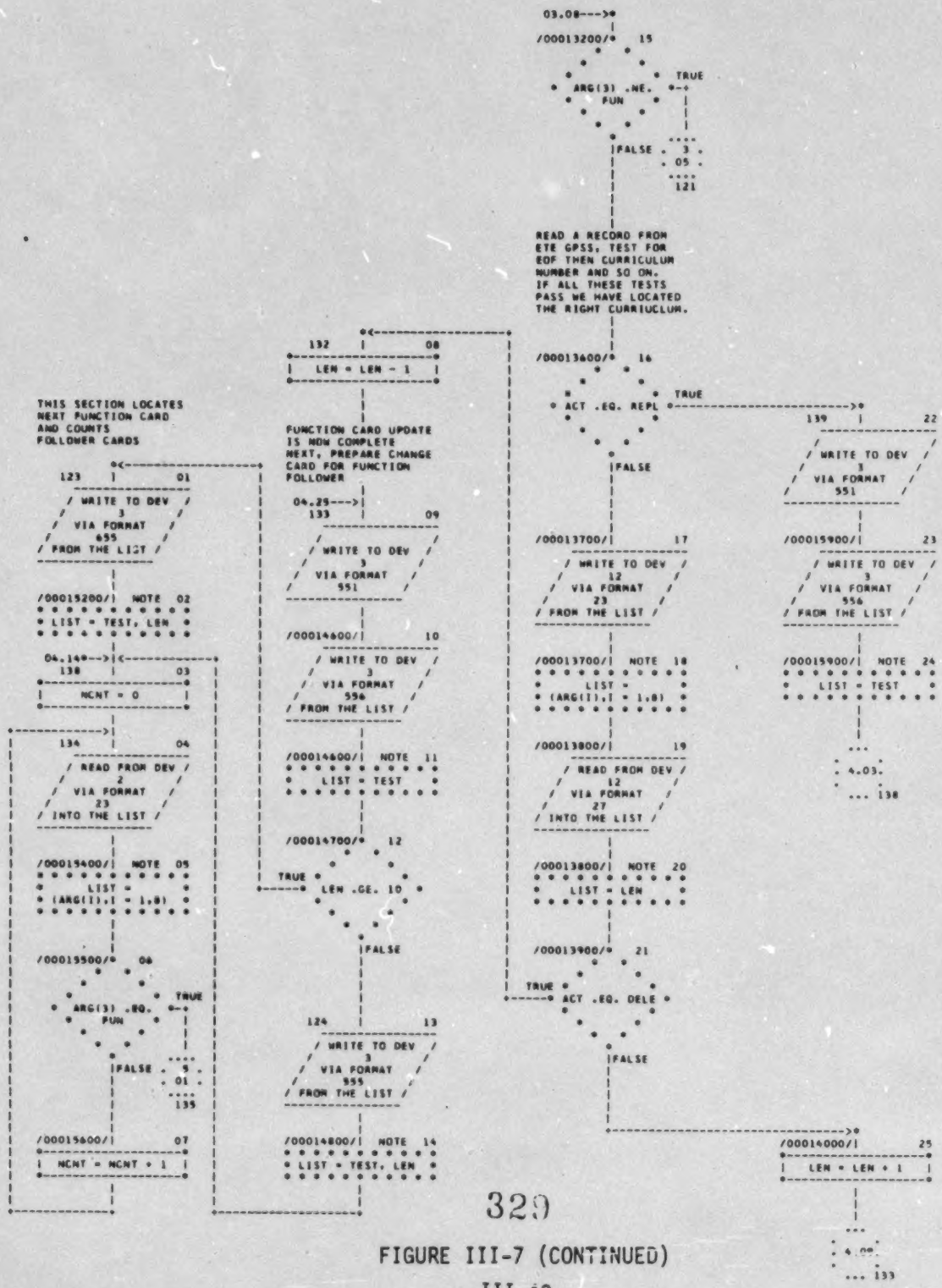
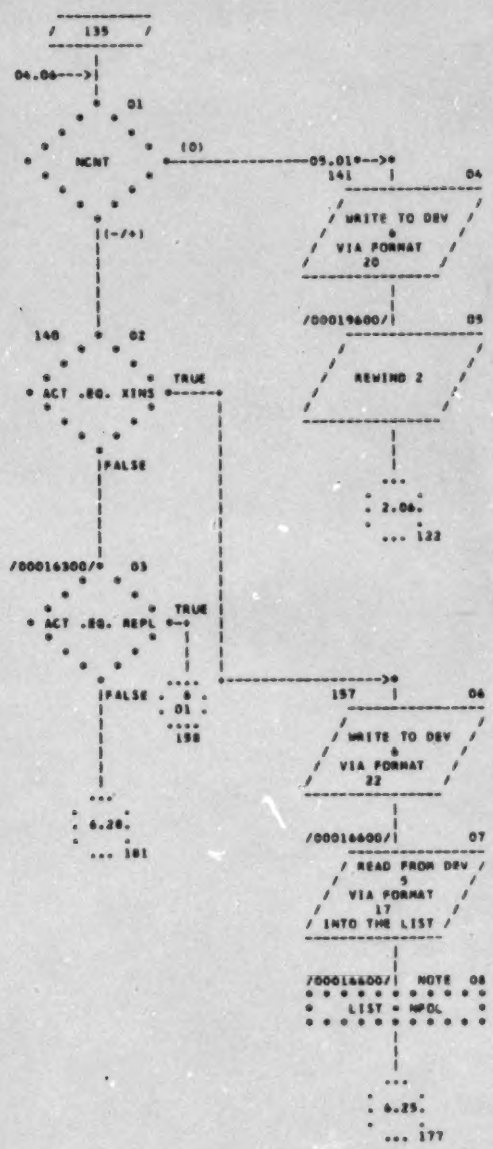


FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES



330

FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES

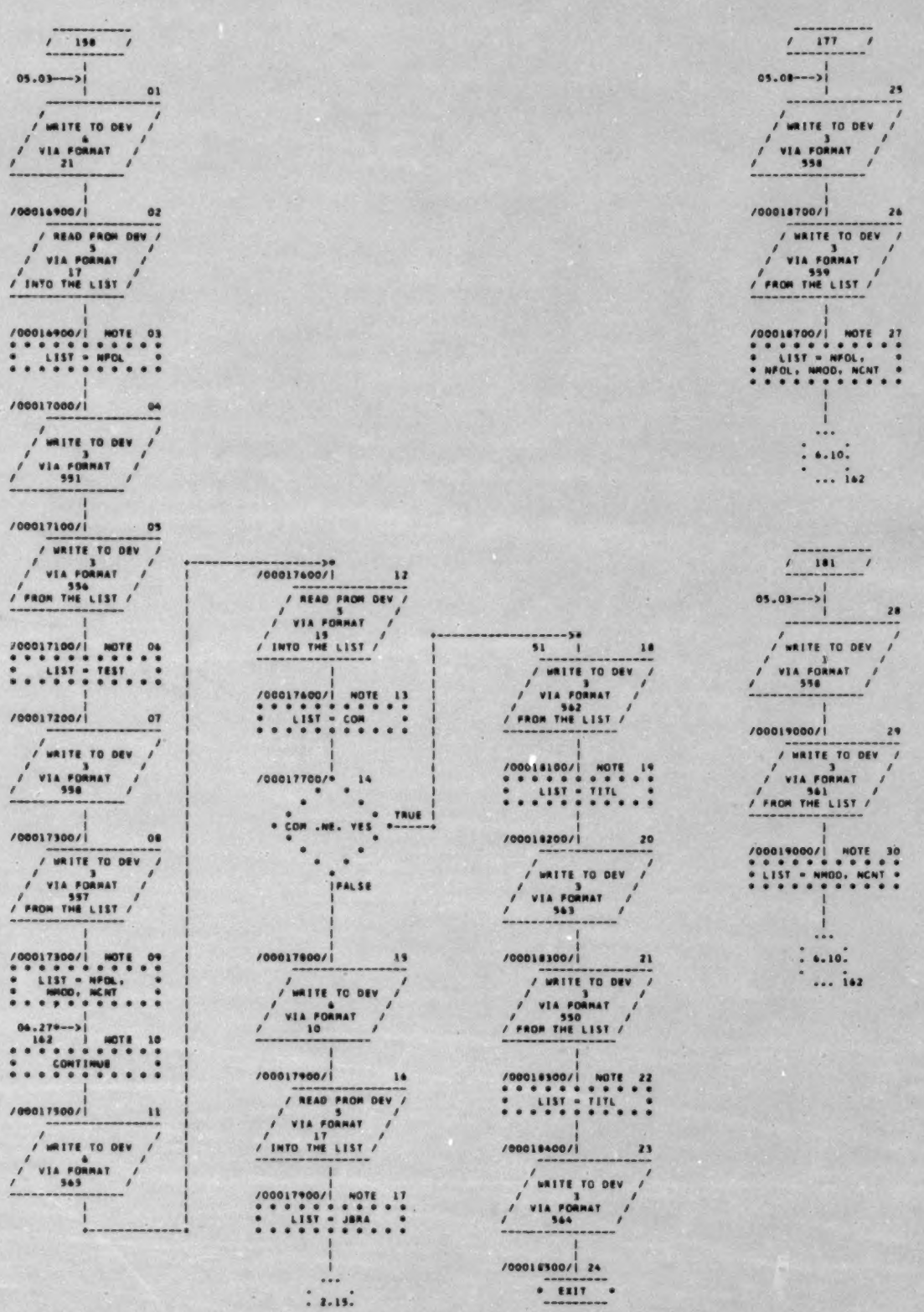


FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES

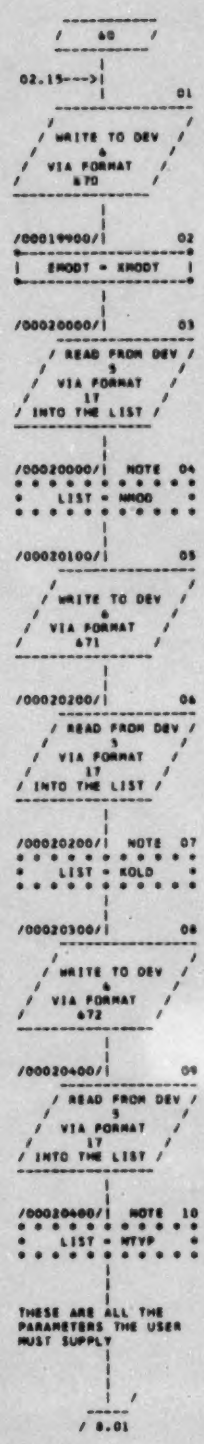


FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES

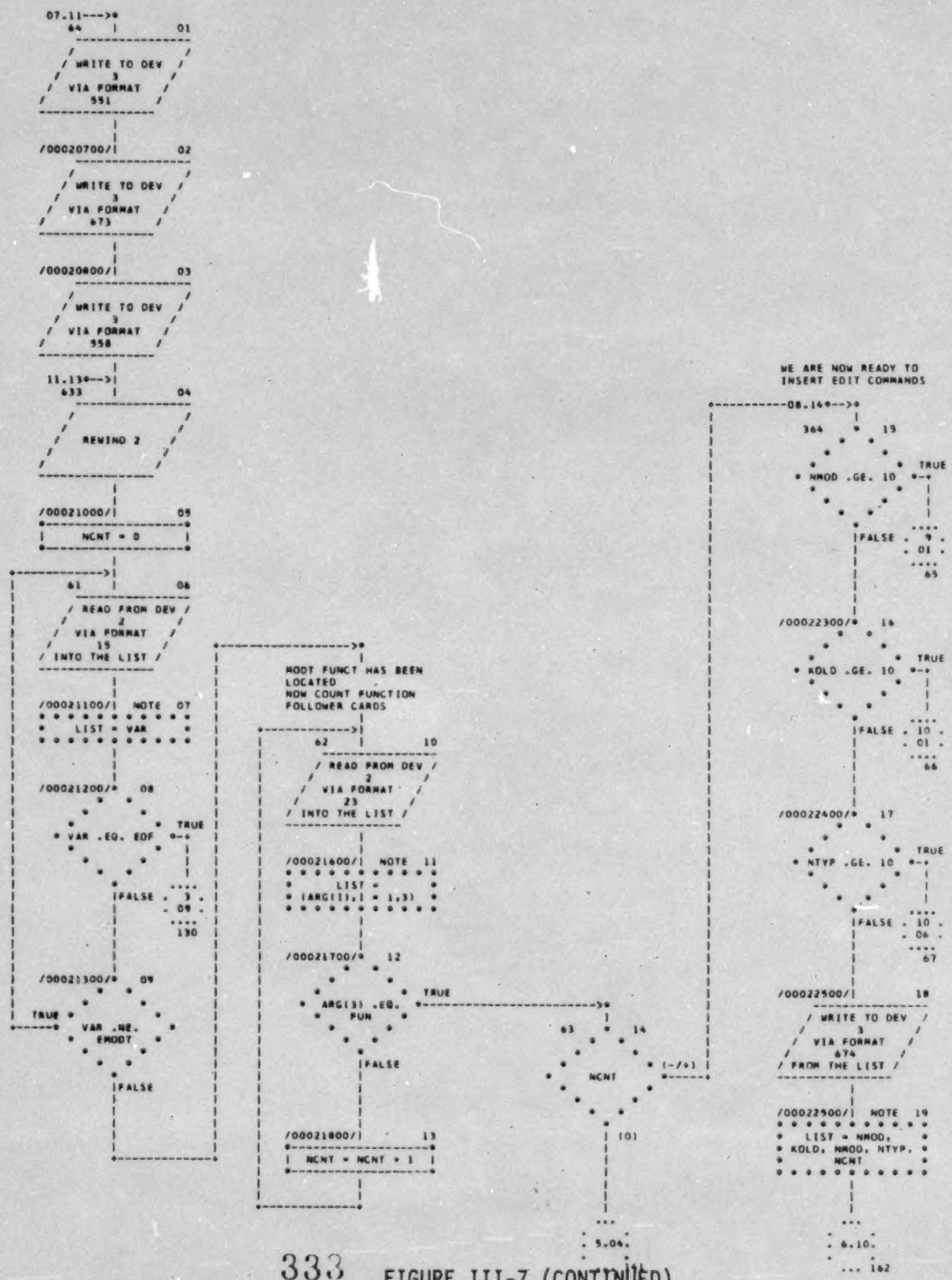


FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES

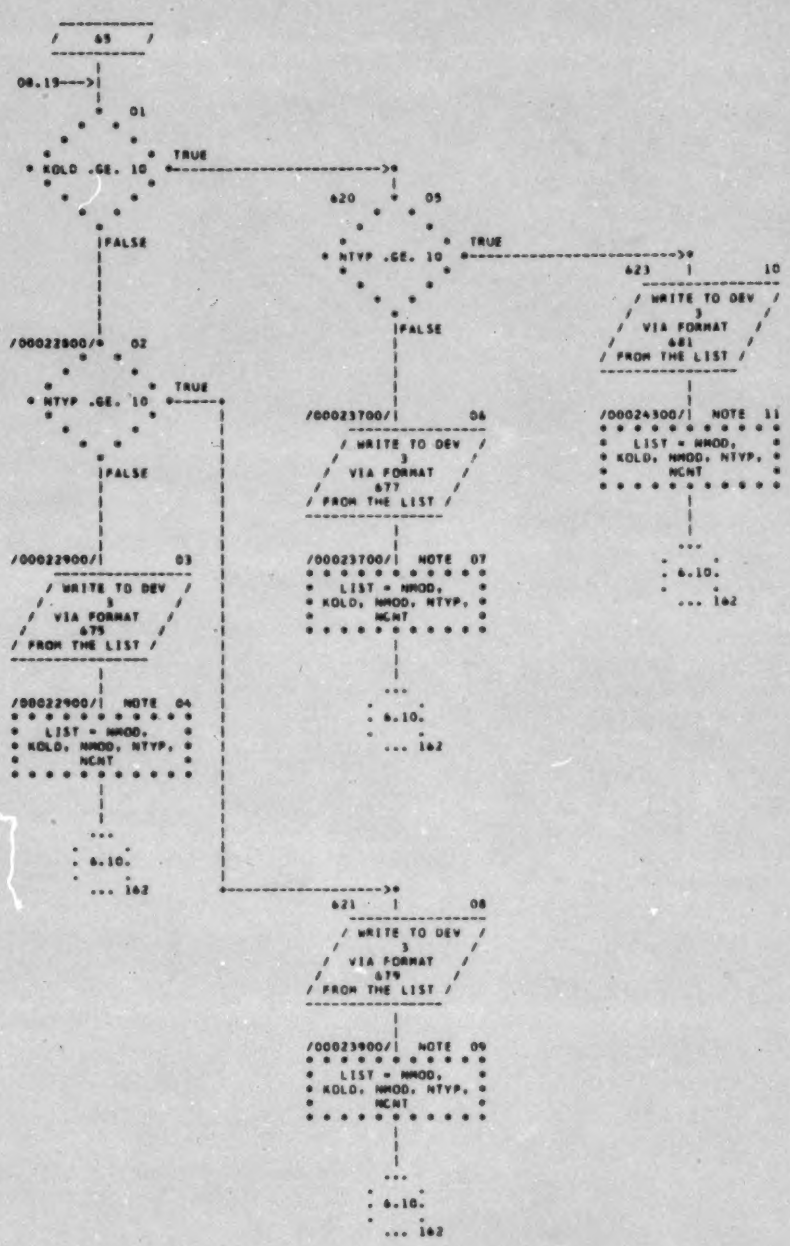


FIGURE III-7 (CONTINUED)

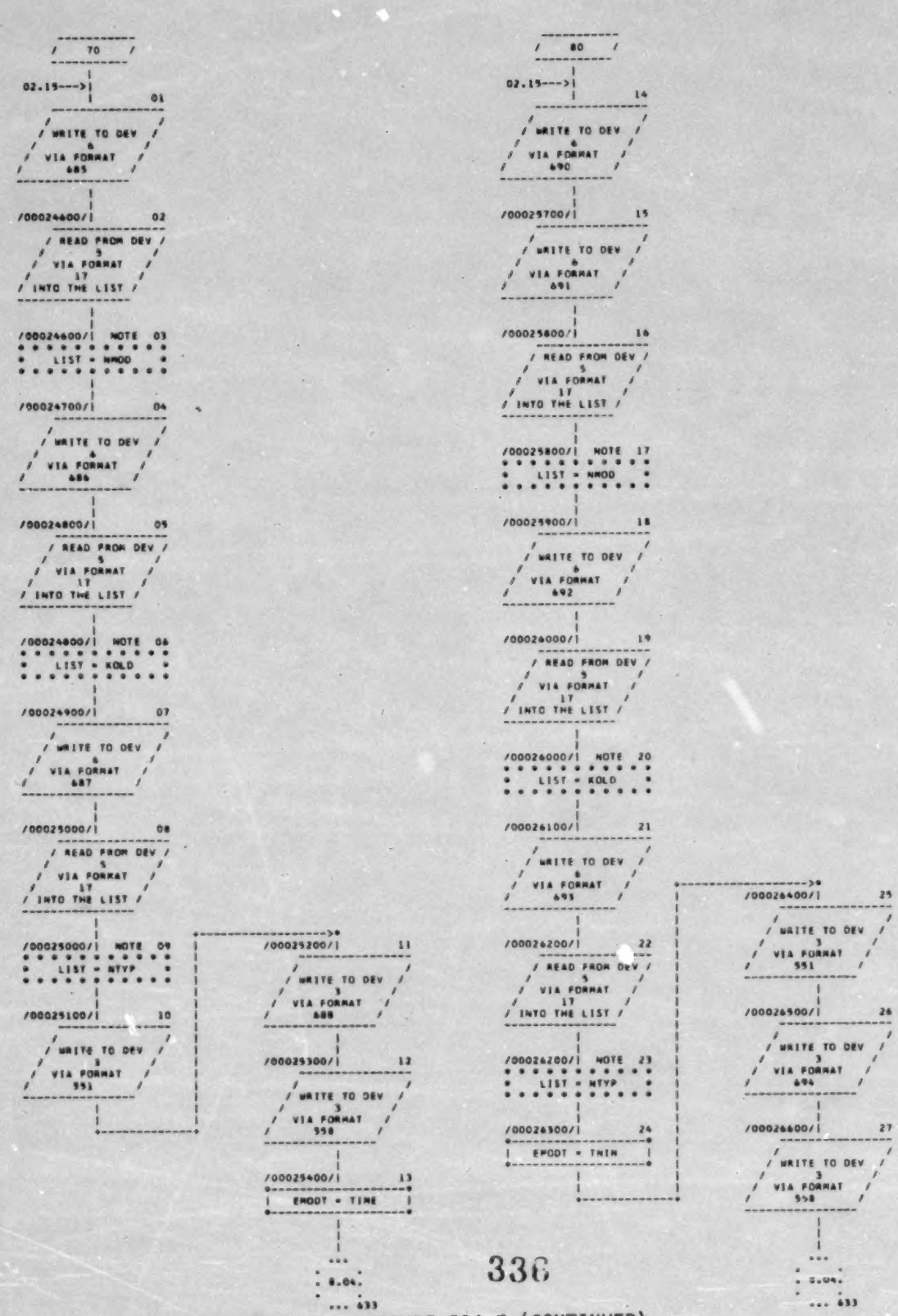
CHART TITLE - PROCEDURES

```
                                                                                  ----------
                                                                                  /   67   /
                                                                                  ----------
                                                                                       |
                                                                             08.17--->|
                                                                                       |          06
                                                                                  ----------------
                                                                                 / WRITE TO DEV  /
                                                                                /       3        /
                                                                               /  VIA FORMAT    /
                                                                              /      678       /
                                                                             / FROM THE LIST /
                                                                             ----------------
                                                                                       |
                                                                                       |
                                                                             /00023400/|  NOTE  07
                                                                             * * * * * * * * * * *
                                                                             *   LIST = NMOD,    *
                                                                             * KOLD, NMOD, NTYP, *
                                                                             *       NCNT        *
                                                                             * * * * * * * * * * *
                                                                                       |
                                                                                       |
                                                                                      ...
                                                                                    .   .
                                                                                    . 6.10.
                                                                                    .   .
                                                                                      ... 162
                  ----------
                  /   66   /
                  ----------
                       |
             08.16--->|
                       |
                       *   01
                     *   *
                   *       *
                  *         *  TRUE
                 * NTYP .GE. 10 *------------------------>*
                  *         *                          622    |        04
                   *       *                                 ----------------
                     *   *                                  / WRITE TO DEV  /
                       *                                   /       3        /
                       |FALSE                             /  VIA FORMAT    /
                       |                                 /      680       /
                       |                                / FROM THE LIST /
                       |                                ----------------
                       |                                         |
                       |                                         |
                       |                                /00024100/|  NOTE  05
                       |                                * * * * * * * * * * *
                       |                                *   LIST = NMOD,    *
                       |                                * KOLD, NMOD, NTYP, *
                       |                                *       NCNT        *
                       |                                * * * * * * * * * * *
                       |                                         |
                       |                                         |
                       |                                         |
                       |                                        ...
                       |                                      .   .
                       |                                      . 6.10.
                       |                                      .   .
                       |                                        ... 162
/00023200/|        02
  ----------------
 / WRITE TO DEV  /
/       3        /
/  VIA FORMAT    /
/      676       /
/ FROM THE LIST /
----------------
         |
         |
/00023200/|  NOTE  03
* * * * * * * * * * *
*   LIST = NMOD,    *
* KOLD, NMOD, NTYP, *
*       NCNT        *
* * * * * * * * * * *
         |
         |
         |
        ...
      .   .
      . 6.10.
      .   .
        ... 162
```

335

FIGURE III-7 (CONTINUED)

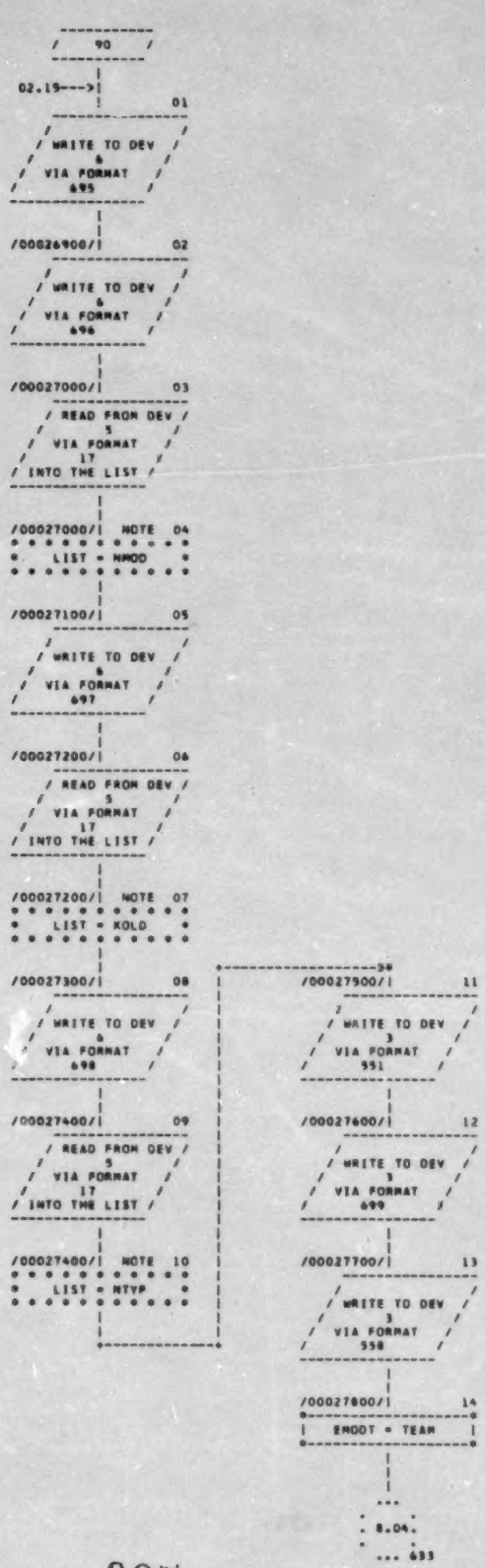
CHART TITLE - PROCEDURES



336

FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES



337 FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES

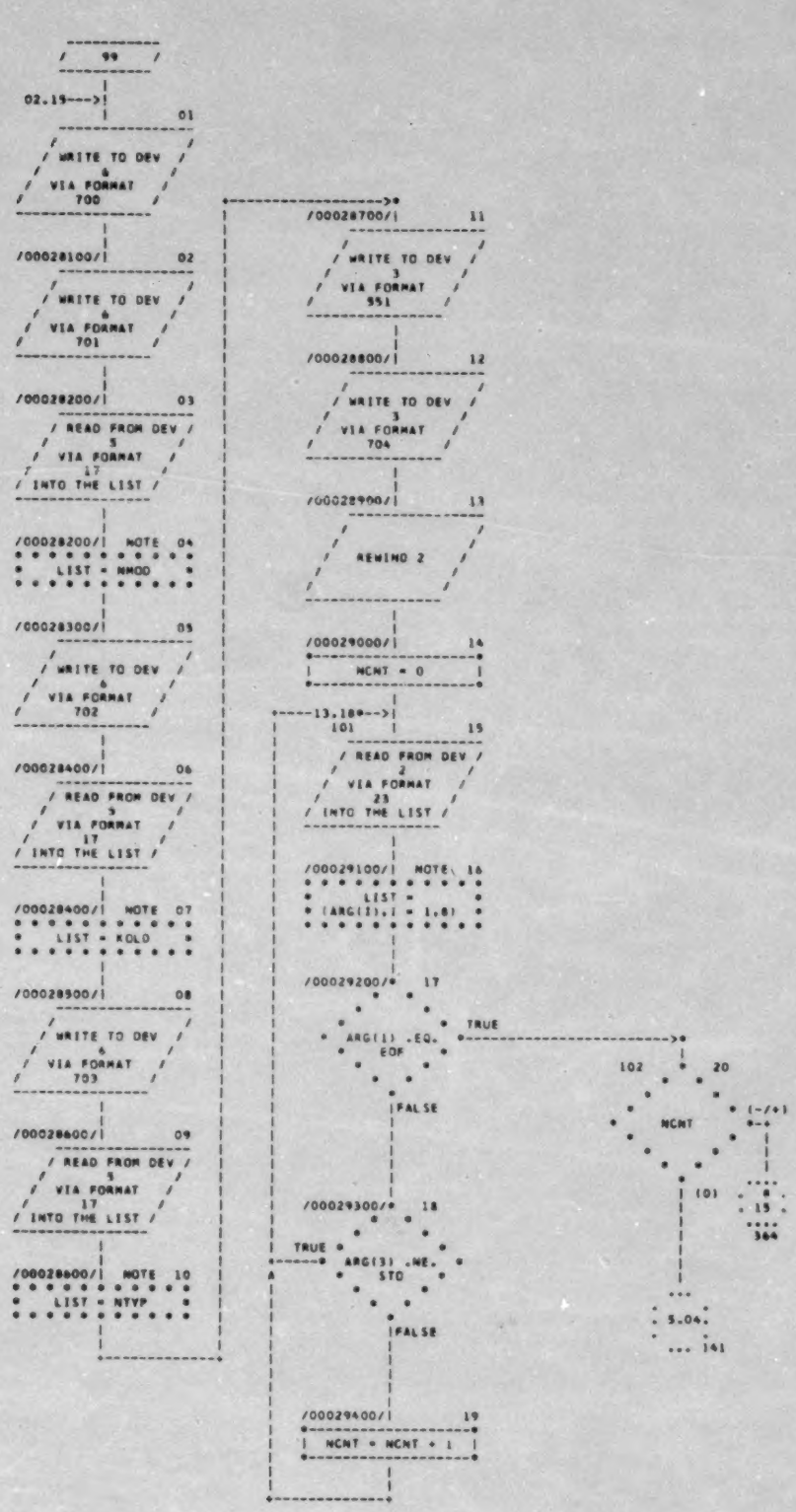
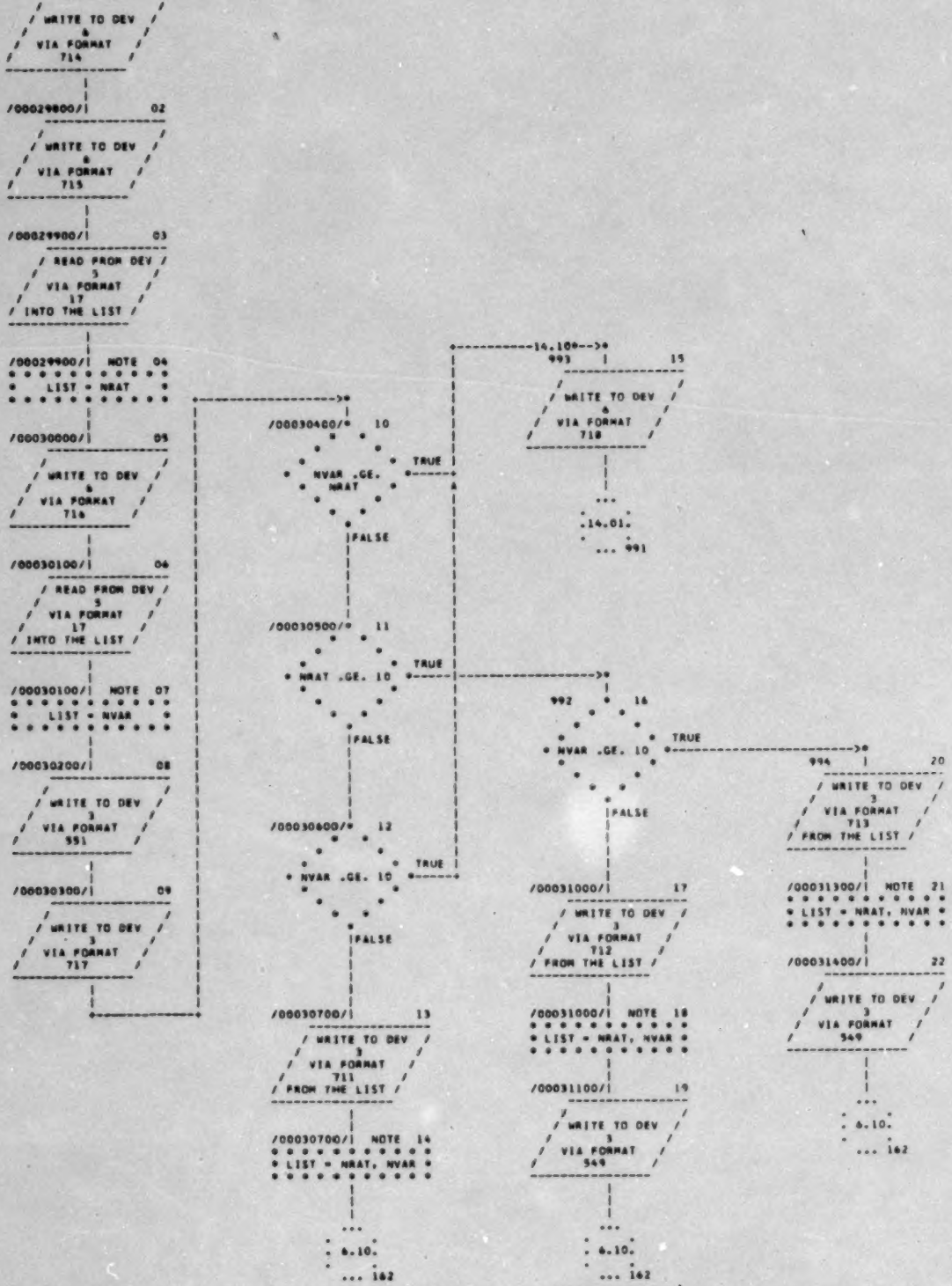


FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES



339 FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES

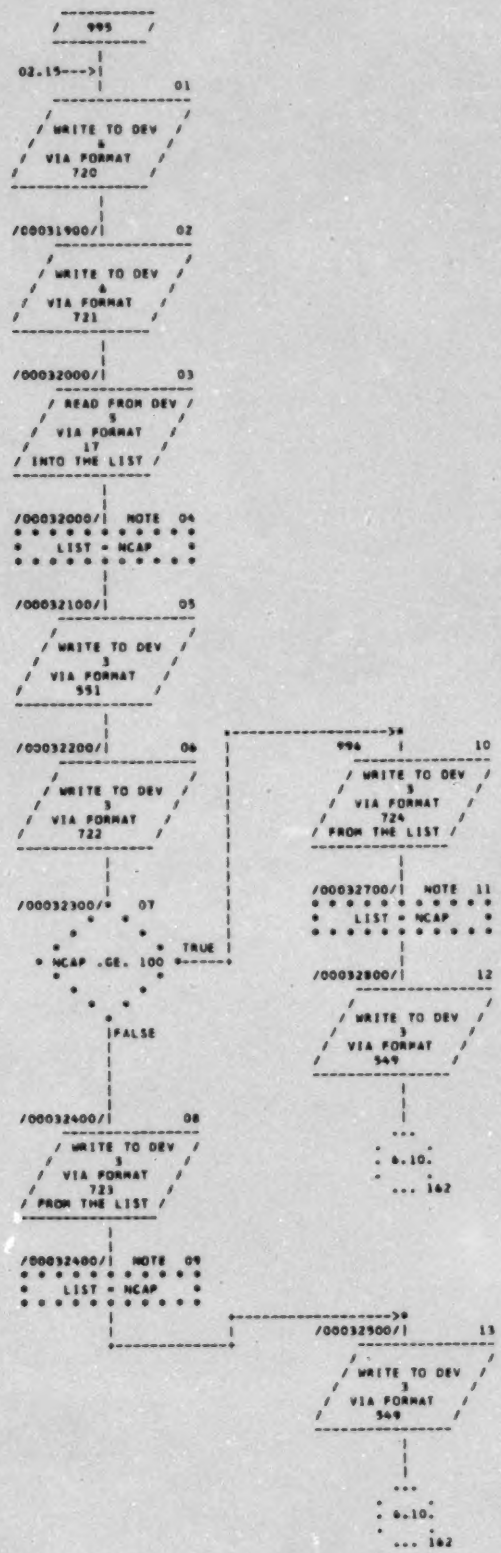


FIGURE III-7 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
900     FORMAT (5I2,A4)
556     FORMAT ('L /',A4,'   FUNC/')
19      FORMAT ('ACTION NOT UNDERSTOOD')
257     FORMAT (1X,I2,1X)
10      FORMAT ('SELECT PARAMETER TO BE CHANGED')
100     FORMAT (' 1 = CURRICULUM')
11      FORMAT ('2 = MODULE TYPE',/,'3 = MODULE COMPLETION TIMES')
12      FORMAT ('4 = NUMBER OF INSTRUCTORS',/,'5 = GROUP SIZES')
13      FORMAT ('6 = FACILITIES')
7       FORMAT  ('7 = STUDENT INPUT RATE')
8       FORMAT('8 = SCHOOL CAPACITY')
29      FORMAT (I1)
14      FORMAT ('ENTER ACTION: INSERT, REPLACE, OR DELETE')
15      FORMAT (A4)
16      FORMAT ('ENTER CURRICULUM NO.')
17      FORMAT (I99)
18      FORMAT ('NEW MODULE = ')
20      FORMAT ('FUNCTION CARD ERROR')
21      FORMAT ('ENTER MOD NO. TO BE REPLACED OR DELETED')
23      FORMAT (8A4)
22      FORMAT ('NEW MODULE TO BE INSERTED FOLLOWING?')
27      FORMAT (22X,I99)
253     FORMAT (I99)
254     FORMAT('CANNOT LOCATE.')
252     FORMAT (I2)
250     FORMAT  (1X,I2,1X)
251     FORMAT (A4)
549     FORMAT ('PRINT 1')
550     FORMAT ('EDIT ',2A4,' GPSS')
551     FORMAT ('TOP')
552     FORMAT ('&BEGSTACK')
553     FORMAT ('FILE',2A4,'GPSS')
554     FORMAT ('&ENDSTACK')
555     FORMAT ('R ',A4,'   FUNCTION   P9,L',I1)
655     FORMAT ('R ',A4,'   FUNCTION   P9,L',I2)
557     FORMAT ('C /,',I1,'/,',I1,'/',I1)
558     FORMAT ('D 1')
559     FORMAT ('C |/,',I1,'|/,',I1,'/,',I1,'|',I1)
561     FORMAT ('C |/,',I1,'||',I1)
562     FORMAT ('FILE ',2A4,'GPSS')
563     FORMAT ('&ENDSTACK')
564      FORMAT ('&EXIT')
560     FORMAT ('&TYPE OFF')
565     FORMAT ('ANYMORE CHANGES?')
670     FORMAT ('MODULE TYPE CODE CHANGE.'/'ENTER MODULE NUMBER')
671     FORMAT ('ENTER OLD TYPE CODE')
672     FORMAT ('ENTER NEW TYPE CODE')
673     FORMAT ('L /MODT/')
```

FIGURE III-7 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
674     FORMAT ('C /',I1,',',I1,'/',I1,',',I1,'/',I1)
675     FORMAT ('C /',I2,',',I1,'/',I2,',',I1,'/',I1)
676     FORMAT ('C /',I1,',',I2,'/',I1,',',I1,'/',I1)
677     FORMAT ('C /',I2,',',I2,'/',I2,',',I1,'/',I1)
678     FORMAT ('C /',I1,',',I1,'/',I1,',',I2,'/',I1)
679     FORMAT ('C /',I2,',',I1,'/',I2,',',I2,'/',I1)
680     FORMAT ('C /',I1,',',I2,'/',I1,',',I2,'/',I1)
681     FORMAT ('C /',I2,',',I2,'/',I2,',',I2,'/',I1)
685     FORMAT ('COMPLETION TIME CHANGE',/,'ENTER MODULE NUMBER')
686     FORMAT ('ENTER OLD COMPLETION TIME')
687     FORMAT ('ENTER NEW COMPLETION TIME')
688     FORMAT ('L /TYNE  F')
690     FORMAT ('CHANGE NUMBER OF INSTRUCTORS')
691     FORMAT ('ENTER INSTRUCTOR GROUP NUMBER')
692     FORMAT ('ENTER OLD GROUP SIZE')
693     FORMAT ('ENTER NEW GROUP SIZE')
694     FORMAT ('L /INSTN F/')
695     FORMAT ('CHANGE STUDENT GROUP SIZE')
696     FORMAT ('ENTER GROUP NUMBER')
697     FORMAT ('ENTER OLD GROUP SIZE')
698     FORMAT ('ENTER NEW GROUP SIZE')
699     FORMAT ('L /TEAM  F/')
700     FORMAT ('CHANGE FACILITIES INVENTORY')
701     FORMAT ('ENTER FACILITY TYPE')
702     FORMAT ('ENTER CURRENT CAPACITY')
703     FORMAT ('ENTER NEW CAPACITY')
704     FORMAT ('L /STORAGE/')
711     FORMAT ('R        GENERATE   ',I1,',',I1,',,,,20')
712     FORMAT ('R        GENERATE   ',I2,',',I1,',,,,20')
713     FORMAT ('R        GENERATE   ',I2,',',I2,',,,,20')
714     FORMAT ('CHANGE STUDENT INPUT RATE')
715     FORMAT ('NEW STUDENT EVERY ? CLOCK UNITS')
716     FORMAT ('ENTER NUMBER OF CLOCK UNITS OF VARIATION')
717     FORMAT ('L /   GENERATE/')
718     FORMAT ('VARIATION GREATER THAN MEAN - TRY AGAIN')
720     FORMAT ('CHANGE STUDENT CAPACITY')
721     FORMAT ('ENTER NEW CAPACITY')
722     FORMAT ('L /AOB   VAR/')
723     FORMAT ('R  AOB   VARIABLE   X',I2)
724     FORMAT ('R  AOB   VARIABLE   X',I3)
        DATA TEAM/' TEA'/
        DATA REPL/'REPL'/,XINS/'INSE'/,DELE/'DELE'/,FUN/'UNCT'/
        DATA EOF/'/*  '/,YES/'YES'/,XMCOT/' MOD'/
        DATA TIME/' TYM'/
        DATA TNIN/' INS'/
        DATA STO/'TORA'/
        DIMENSION ARG (8),TITL(2)
```

FIGURE III-7 (CONTINUED)

CHART TITLE - PROCEDURES

```
 -----------
//00033400//
-----------
     |
     |
     | 01
 ---------
*  EXIT   *
 ---------

   RETURN
BE SURE THAT ALL
STORAGE CARDS ARE
   CHECKED.
```

343

FIGURE III-7 (CONTINUED)

INPUT LISTING AUTOFLOW CHART SET - FINAL -MAD- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSD,SOURCSEQ)

```
CARD NO    ****                    CONTENTS                                  ****

  1        900  FORMAT (5I2,A4)                                                00000100
  2        994  FORMAT ('L /',A4,'   FUNC/')                                   00000200
  3         19  FORMAT ('ACTION NOT UNDERSTOOD')                               00000300
  4        257  FORMAT (1X,I2,1X)                                              00000400
  5         10  FORMAT ('SELECT PARAMETER TO BE CHANGED')                      00000500
  6        100  FORMAT (' 1 = CURRICULUM')                                     00000600
  7         11  FORMAT ('2 = MODULE TYPE',/,'3 = MODULE COMPLETION TIMES')     00000700
  8         12  FORMAT ('4 = NUMBER OF INSTRUCTORS',/,'5 = GROUP SIZES')       00000800
  9         13  FORMAT ('6 = FACILITIES')                                      00000900
 10          7  FORMAT  ('7 = STUDENT INPUT RATE')                             00001000
 11          8  FORMAT('8 = SCHOOL CAPACITY')                                 00001100
 12         29  FORMAT (I1)                                                    00001200
 13         14  FORMAT ('ENTER ACTION: INSERT, REPLACE, OR DELETE')            00001300
 14         15  FORMAT (A4)                                                    00001400
 15         16  FORMAT ('ENTER CURRICULUM NO.')                                00001500
 16         17  FORMAT (I99)                                                   00001600
 17         18  FORMAT ('NEW MODULE = ')                                       00001700
 18         20  FORMAT ('FUNCTION CARD ERROR')                                 00001800
 19         21  FORMAT ('ENTER MOD NO. TO BE REPLACED OR DELETED')             00001900
 20         23  FORMAT (8A4)                                                   00002000
 21         22  FORMAT ('NEW MODULE TO BE INSERTED FOLLOWING?')                00002100
 22         27  FORMAT (22X,I99)                                               00002200
 23        253  FORMAT (I99)                                                   00002300
 24        254  FORMAT('CANNOT LOCATE.')                                       00002400
 25        252  FORMAT (I2)                                                    00002500
 26        250  FORMAT  (1X,I2,1X)                                             00002600
 27        251  FORMAT (A4)                                                    00002700
 28        549  FORMAT ('PRINT 1')                                             00002800
 29        550  FORMAT ('EDIT ',2A4,' GPSS')                                   00002900
 30        551  FORMAT ('TOP')                                                 00003000
 31        552  FORMAT ('&BEGSTACK')                                           00003100
 32        553  FORMAT ('FILE',2A4,'GPSS')                                     00003200
 33        554  FORMAT ('&ENDSTACK')                                           00003300
 34        555  FORMAT ('R ',A4,'   FUNCTION   P9,L',I1)                       00003400
 35        655  FORMAT ('R ',A4,'   FUNCTION   P9,L',I1)                       00003500
 36        557  FORMAT ('C /,',I1,'/,',I1,'/',I1)                              00003600
 37        558  FORMAT ('D 1')                                                 00003700
 38        559  FORMAT ('C 1/-',I1,'1/,',I1,'/,',I1,'1',I1)                    00003800
 39        561  FORMAT ('C 1/,',I1,'11',I1)                                    00003900
 40        562  FORMAT ('FILE ',2A4,'GPSS')                                    00004000
 41        563  FORMAT ('&ENDSTACK')                                           00004100
 42        564   FORMAT ('&EXIT')                                              00004200
 43        560  FORMAT ('&TYPE OFF')                                           00004300
 44        565  FORMAT ('ANYMORE CHANGES?')                                    00004400
 45        670  FORMAT ('MODULE TYPE CODE CHANGE.'/'ENTER MODULE NUMBER')      00004500
 46        671  FORMAT ('ENTER OLD TYPE CODE')                                 00004600
 47        672  FORMAT ('ENTER NEW TYPE CODE')                                 00004700
 48        673  FORMAT ('L /MODT/')                                            00004800
 49        674  FORMAT ('C /',I1,',',I1,'/',I1,',',I1,'/',I1)                  00004900
 50        675  FORMAT ('C /',I2,',',I1,'/',I2,',',I1,'/',I1)                  00005000
 51        676  FORMAT ('C /',I1,',',I2,'/',I1,',',I1,'/',I1)                  00005100
 52        677  FORMAT ('C /',I2,',',I2,'/',I2,',',I1,'/',I1)                  00005200
 53        678  FORMAT ('C /',I1,',',I1,'/',I1,',',I2,'/',I1)                  00005300
 54        679  FORMAT ('C /',I2,',',I1,'/',I2,',',I2,'/',I1)                  00005400
 55        680  FORMAT ('C /',I1,',',I2,'/',I1,',',I2,'/',I1)                  00005500
 56        681  FORMAT ('C /',I2,',',I2,'/',I2,',',I2,'/',I1)                  00005600
 57        685  FORMAT ('COMPLETION TIME CHANGE',/,'ENTER MODULE NUMBER')      00005700
 58        686  FORMAT ('ENTER OLD COMPLETION TIME')                          00005800
 59        687  FORMAT ('ENTER NEW COMPLETION TIME')                           00005900
 60        688  FORMAT ('L /TYME  F')                                          00006000
 61        690  FORMAT ('CHANGE NUMBER OF INSTRUCTORS')                       00006100
 62        691  FORMAT ('ENTER INSTRUCTOR GROUP NUMBER')                      00006200
 63        692  FORMAT ('ENTER OLD GROUP SIZE')                                00006300
 64        693  FORMAT ('ENTER NEW GROUP SIZE')                                00006400
 65        694  FORMAT ('L /INSTN F/')                                         00006500
 66        695  FORMAT ('CHANGE STUDENT GROUP SIZE')                           00006600
 67        696  FORMAT ('ENTER GROUP NUMBER')                                  00006700
 68        697  FORMAT ('ENTER OLD GROUP SIZE')                                00006800
 69        698  FORMAT ('ENTER NEW GROUP SIZE')                                00006900
 70        699  FORMAT ('L /TEAM  F/')                                         00007000
 71        700  FORMAT ('CHANGE FACILITIES INVENTORY')                        00007100
 72        701  FORMAT ('ENTER FACILITY TYPE')                                00007200
 73        702  FORMAT ('ENTER CURRENT CAPACITY')                             00007300
 74        703  FORMAT ('ENTER NEW CAPACITY')                                 00007400
 75        704  FORMAT ('L /STORAGE/')                                        00007500
 76        711  FORMAT ('R        GENERATE   ',I1,',',I1,',,,,20')            00007600
 77        712  FORMAT ('R        GENERATE   ',I2,',',I1,',,,,20')            00007700
 78        713  FORMAT ('R        GENERATE   ',I2,',',I2,',,,,20')            00007800
 79        714  FORMAT ('CHANGE STUDENT INPUT RATE')                           00007900
 80        715  FORMAT ('NEW STUDENT EVERY 7 CLOCK UNITS')                     00008000
 81        716  FORMAT ('ENTER NUMBER OF CLOCK UNITS OF VARIATION')            00008100
 82        717  FORMAT ('L /   GENERATE/')                                     00008200
 83        718  FORMAT ('VARIATION GREATER THAN MEAN - TRY AGAIN')             00008300
 84         720 FORMAT ('CHANGE STUDENT CAPACITY')                             00008400
 85         721 FORMAT ('ENTER NEW CAPACITY')                                  00008500
 86         722 FORMAT ('L /ADB   VAR/')                                       00008600
 87         723 FORMAT ('R  ADB   VARIABLE   K',I2)                            00008700
 88         724 FORMAT ('R  ADB   VARIABLE   K',I3)                            00008800
 89             DATA TEAM/' TEA'/                                              00008900
 90             DATA REPL/'REPL'/,XINS/'INSE'/,DELE/'DELE'/,FUN/'UNCT'/        00009000
 91             DATA EOF/'/*  '/,YES/'YES'/,XMODT/' MOD'/                      00009100
 92             DATA TIME/' TYM'/                                              00009200
 93             DATA TNIN/' INS'/                                              00009300
 94             DATA STO/'TORA'/                                               00009400
 95             DIMENSION ARG (8),TITL(2)                                      00009500
 96       C THIS SET OF WRITES INITIALIZES THE EXEC: &TYPE OFF ETC.            00009600
 97             READ (5,23)TITL                                                00009700
 98             WRITE (3,560)                                                  00009800
 99             WRITE (3,552)                                                  00009900
100             CALL REREAD (12)                                               00010000
101        122  WRITE (6,10)                                                   00010100
102             WRITE (6,100)                                                  00010200
103             WRITE (6,11)                                                   00010300
104             WRITE (6,12)                                                   00010400
105             WRITE (6,13)                                                   00010500
106             WRITE (6,7)                                                    00010600
107             WRITE (6,8)                                                    00010700
108             READ (5,29)JBRA                                                00010800
109        323  GOTO (50,60,70,80,90,99,991,995),JBRA                         00010900
```

TABLE III-5

INPUT LISTING AUTOFLOW CHART SET - FINAL -MAD- FLOWCHARTS

CARD NO **** CONTENTS ****

```
C THIS SECTION DEALS WITH CURRICULUM UPDATE                          00011000
 50   WRITE (6,14)                                                   00011100
      READ (5,15) ACT                                                00011200
      REWIND 2                                                       00011300
      WRITE (6,16)                                                   00011400
      READ (5,17)NCUR                                                00011500
C FOR REPLACE THERE IS NO NEED TO CHANGE FUNCTION CARD                00011600
      IF(ACT.EQ.DELE)GOTO 52                                         00011700
      WRITE (6,18)                                                   00011800
 154  READ (5,17)NMOD                                                00011900
      IF(NCUR.GE.10) GOTO 120                                        00012000
      WRITE (12,250)NCUR                                             00012100
      GOTO 125                                                       00012200
 52   WRITE (6,21)                                                   00012300
      GOTO 154                                                       00012400
 120  WRITE (12,257)NCUR                                             00012500
 125  READ(12,251)TEST                                               00012600
C  FILE: TURK SHOULD BE SET UP AS SOON AS PROGRAM ENTERED             00012700
C &BEGSTACK AND EDIT REQUEST ENTERED INTO IT.                         00012800
 121  READ (2,23)(ARG(I),I=1,8)                                      00012900
      IF(ARG(1).EQ.EOF)GOTO 130                                      00013000
      IF(ARG(1).NE.TEST)GOTO 121                                     00013100
      IF(ARG(3).NE.FUN) GOTO 121                                     00013200
C READ A RECORD FROM ETE GPSS, TEST FOR EOF THEN CURRICULUM           00013300
C  NUMBER AND SO ON.                                                 00013400
C IF ALL THESE TESTS PASS WE HAVE LOCATED THE RIGHT CURRIUCLUM.       00013500
      IF(ACT.EQ.REPL) GOTO 139                                       00013600
      WRITE (12,23)(ARG(I),I=1,8)                                    00013700
      READ (12,27)LEN                                                00013800
      IF(ACT.EQ.DELE)GOTO 132                                        00013900
      LEN = LEN+ 1                                                   00014000
      GOTO 133                                                       00014100
 132  LEN = LEN-1                                                    00014200
C FUNCTION CARD UPDATE IS NOW COMPLETE                                00014300
C NEXT, PREPARE CHANGE CARD FOR FUNCTION FOLLOWER                     00014400
 133  WRITE (3,551)                                                  00014500
      WRITE (3,556)TEST                                              00014600
      IF (LEN.GE.10) GOTO 123                                        00014700
 124  WRITE (3,555)TEST,LEN                                          00014800
      GOTO 138                                                       00014900
C THIS SECTION LOCATES NEXT FUNCTION CARD AND COUNTS                  00015000
C  FOLLOWER CARDS                                                    00015100
 123  WRITE (3,655)TEST,LEN                                          00015200
 138  NCNT = 0                                                       00015300
 134  READ (2,23)(ARG(I),I = 1,8)                                    00015400
      IF (ARG(3).EQ.FUN)GOTO 135                                     00015500
      NCNT = NCNT +1                                                 00015600
      GOTO 134                                                       00015700
 139 WRITE (3,551)                                                   00015800
      WRITE (3,556)TEST                                              00015900
      GOTO 138                                                       00016000
 135  IF(NCNT) 140,141,140                                           00016100
 140  IF(ACT.EQ.XINS)GOTO 157                                        00016200
      IF(ACT.EQ.REPL) GOTO 158                                       00016300
      GOTO 181                                                       00016400
 157  WRITE (6,22)                                                   00016500
      READ (5,17)NFOL                                                00016600
      GOTO 177                                                       00016700
 158  WRITE (6,21)                                                   00016800
      READ (5,17)NFOL                                                00016900
      WRITE (3,551)                                                  00017000
      WRITE (3,556)TEST                                              00017100
      WRITE (3,558)                                                  00017200
      WRITE (3,557)NFOL,NMOD,NCNT                                    00017300
 162  CONTINUE                                                       00017400
      WRITE (6,565)                                                  00017500
      READ (5,15)COM                                                 00017600
      IF (COM.NE.YES)GOTO 51                                         00017700
      WRITE (6,10)                                                   00017800
      READ (5,17)JBRA                                                00017900
      GOTO 323                                                       00018000
 51   WRITE (3,562)TITL                                              00018100
      WRITE (3,563)                                                  00018200
      WRITE (3,550)TITL                                              00018300
      WRITE (3,564)                                                  00018400
      CALL EXIT                                                      00018500
 177  WRITE (3,558)                                                  00018600
      WRITE (3,559)NFOL,NFOL,NMOD,NCNT                               00018700
      GOTO 162                                                       00018800
 181  WRITE (3,558)                                                  00018900
      WRITE (3,561)NMOD,NCNT                                         00019000
      GOTO 162                                                       00019100
 130  WRITE (6,254)                                                  00019200
      REWIND 2                                                       00019300
      GOTO 122                                                       00019400
 141  WRITE (6,20)                                                   00019500
      REWIND 2                                                       00019600
      GOTO 122                                                       00019700
 60   WRITE (6,670)                                                  00019800
      EMODT = XMODT                                                  00019900
      READ (5,17)NMOD                                                00020000
      WRITE (6,671)                                                  00020100
      READ (5,17)KOLD                                                00020200
      WRITE (6,672)                                                  00020300
      READ (5,17)NTYP                                                00020400
C THESE ARE ALL THE PARAMETERS THE USER MUST SUPPLY                   00020500
 64   WRITE (3,551)                                                  00020600
      WRITE (3,673)                                                  00020700
      WRITE (3,558)                                                  00020800
 633  REWIND 2                                                       00020900
      NCNT = 0                                                       00021000
  61  READ (2,15)VAR                                                 00021100
      IF (VAR.EQ.EOF)GOTO 130                                        00021200
      IF (VAR.NE.EMODT) GOTO 61                                      00021300
C MODT FUNCT HAS BEEN LOCATED                                         00021400
C NOW COUNT FUNCTION FOLLOWER CARDS                                   00021500
  62  READ (2,23)(ARG (I),I = 1,3)                                    00021600
         IF (ARG(3).EQ.FUN)GOTO 63                                   00021700
      NCNT = NCNT+1                                                  00021800
      GOTO 62                                                        00021900
  63  IF(NCNT)364,141,364                                            00022000
C WE ARE NOW READY TO INSERT EDIT COMMANDS                            00022100
```

345 TABLE III-5 (CONTINUED)

INPUT LISTING — AUTOFLOW CHART SET - FINAL — -MAD- FLOWCHARTS

```
CARD NO   ****   CONTENTS                                          ****

  222     364    IF(NMOD.GE.10)GOTO 65                             00022200
  223            IF(KOLD.GE.10)GOTO 66                             00022300
  224            IF(NTYP.GE.10)GOTO 67                             00022400
  225            WRITE (3,674)NMOD,KOLD,NMOD,NTYP,NCNT             00022500
  226            GOTO 162                                          00022600
  227     65     IF(KOLD.GE.10)GOTO 620                            00022700
  228            IF(NTYP.GE.10)GOTO 621                            00022800
  229            WRITE (3,675)NMOD,KOLD,NMOD,NTYP,NCNT             00022900
  230            GOTO 162                                          00023000
  231     66     IF(NTYP.GE.10)GOTO 622                            00023100
  232            WRITE (3,676)NMOD,KOLD,NMOD,NTYP,NCNT             00023200
  233            GOTO 162                                          00023300
  234     67     WRITE (3,678)NMOD,KOLD,NMOD,NTYP,NCNT             00023400
  235            GOTO 162                                          00023500
  236     620    IF(NTYP.GE.10)GOTO 623                            00023600
  237            WRITE (3,677)NMOD,KOLD,NMOD,NTYP,NCNT             00023700
  238            GOTO 162                                          00023800
  239     621    WRITE (3,679)NMOD,KOLD,NMOD,NTYP,NCNT             00023900
  240            GOTO 162                                          00024000
  241     622    WRITE (3,680)NMOD,KOLD,NMOD,NTYP,NCNT             00024100
  242            GOTO 162                                          00024200
  243     623    WRITE (3,681)NMOD,KOLD,NMOD,NTYP,NCNT             00024300
  244            GOTO 162                                          00024400
  245     70     WRITE (6,685)                                     00024500
  246            READ (5,17)NMOD                                   00024600
  247            WRITE (6,686)                                     00024700
  248            READ (5,17)KOLD                                   00024800
  249            WRITE (6,687)                                     00024900
  250            READ (5,17)NTYP                                   00025000
  251            WRITE (3,551)                                     00025100
  252            WRITE (3,688)                                     00025200
  253            WRITE (3,558)                                     00025300
  254            EMODT = TIME                                      00025400
  255            GOTO 633                                          00025500
  256     80     WRITE (6,690)                                     00025600
  257            WRITE (6,691)                                     00025700
  258            READ (5,17)NMOD                                   00025800
  259            WRITE (6,692)                                     00025900
  260            READ (5,17)KOLD                                   00026000
  261            WRITE (6,693)                                     00026100
  262            READ (5,17)NTYP                                   00026200
  263            EMODT = TNIN                                      00026300
  264            WRITE (3,551)                                     00026400
  265            WRITE (3,694)                                     00026500
  266            WRITE (3,558)                                     00026600
  267            GOTO 633                                          00026700
  268     90     WRITE (6,695)                                     00026800
  269            WRITE (6,696)                                     00026900
  270            READ (5,17)NMOD                                   00027000
  271            WRITE (6,697)                                     00027100
  272            READ (5,17)KOLD                                   00027200
  273            WRITE (6,698)                                     00027300
  274            READ (5,17)NTYP                                   00027400
  275            WRITE (3,551)                                     00027500
  276            WRITE (3,699)                                     00027600
  277            WRITE (3,558)                                     00027700
  278            EMODT = TEAM                                      00027800
  279            GOTO 633                                          00027900
  280     99     WRITE (6,700)                                     00028000
  281            WRITE (6,701)                                     00028100
  282            READ (5,17)NMOD                                   00028200
  283            WRITE (6,702)                                     00028300
  284            READ (5,17)KOLD                                   00028400
  285            WRITE (6,703)                                     00028500
  286            READ (5,17)NTYP                                   00028600
  287            WRITE (3,551)                                     00028700
  288            WRITE (3,704)                                     00028800
  289            REWIND 2                                          00028900
  290            NCNT = 0                                          00029000
  291     101    READ (2,23)(ARG(I),I= 1,8)                        00029100
  292            IF (ARG(1).EQ.EOF) GOTO 102                       00029200
  293            IF (ARG(3).NE.STO) GOTO 101                       00029300
  294            NCNT = NCNT +1                                    00029400
  295            GOTO 101                                          00029500
  296     102    IF (NCNT) 364,141,364                             00029600
  297     991    WRITE (6,714)                                     00029700
  298            WRITE (6,715)                                     00029800
  299            READ (5,17)NRAT                                   00029900
  300            WRITE (6,716)                                     00030000
  301            READ (5,17)NVAR                                   00030100
  302            WRITE (3,551)                                     00030200
  303            WRITE (3,717)                                     00030300
  304            IF (NVAR.GE.NRAT)GOTO 993                         00030400
  305            IF(NRAT.GE.10)GOTO 992                            00030500
  306            IF(NVAR.GE.10)GOTO 995                            00030600
  307            WRITE (3,711)NRAT,NVAR                            00030700
  308            GOTO 162                                          00030800
  309     992    IF(NVAR.GE.10)GOTO 994                            00030900
  310            WRITE (3,712)NRAT,NVAR                            00031000
  311            WRITE (3,549)                                     00031100
  312            GOTO 162                                          00031200
  313     994    WRITE (3,713)NRAT,NVAR                            00031300
  314            WRITE (3,549)                                     00031400
  315            GOTO 162                                          00031500
  316     993    WRITE (6,718)                                     00031600
  317            GOTO 991                                          00031700
  318     995    WRITE (6,720)                                     00031800
  319            WRITE (6,721)                                     00031900
  320            READ (5,17)NCAP                                   00032000
  321            WRITE (3,551)                                     00032100
  322            WRITE (3,722)                                     00032200
  323            IF(NCAP.GE.100) GOTO 996                          00032300
  324            WRITE (3,723)NCAP                                 00032400
  325            WRITE(3,549)                                      00032500
  326            GOTO 162                                          00032600
  327     996    WRITE (3,724)NCAP                                 00032700
  328            WRITE (3,549)                                     00032800
  329            GOTO 162                                          00032900
  330            END                                               00033000
  331     C FOR CHANGES INVOLVING STORAGES THE ASSUMPTION IS       00033100
  332     C  MADE THAT THEY ARE SPECIFIED INDIVIDUALLY.            00033200
  333     C    HOWEVER, A COUNT OF STORAGE CARDS WILL BE MADE TO   00033300
  334     C     BE SURE THAT ALL STORAGE CARDS ARE CHECKED.        00033400
```

TABLE III-5 (CONTINUED)

SIMULATION SUBSYSTEM

The simulation subsystem consists of one GPSS program (DUMMY) and its associated executive (GPSS EXEC). As previously mentioned, DUMMY does not consitute an executable model. Only after insertion of the required FUNCTION and VARIABLE statements is the model executable.

GPSS EXEC forms the link between ETE2 and the NGPSS compiler. It is transparent to the user and requires no operator intervention. GPSS EXEC provides responses to the queries of the NGPSS compiler via the STACK command. GPSS EXEC passes the name of the model to be executed to the NGPSS system and causes NGPSS to terminate execution after a single model run.

DUMMY GPSS contains all the simulation logic. Function cards are inserted beginning at the *START card. This area is clearly denoted in the program listing by a set of comments cards. The functions supplied by the model generation subsystem are listed and described in Figure III-8.

Figure III- 9 is a flow of the logic contained in DUMMY GPSS. Model logic can be summarized as follows:

1. Transactions (students) are generated at a fixed rate.

2. Each time a transaction is generated, a test is made to see if the system already contains the maximum permissible number of transactions (a variable designated AOB is used for this test). The total number of transactions currently in the system is contained in SAVEVALUE 101. If the system is full (i.e. SAVEVALUE 101 AOB) the transaction is destroyed.

3. If the transaction is permitted to enter the system, it is then assigned a transaction type number via the function INTIM. Once the type number has been assigned, it remains the same as long as the transaction is in the system.

4. For each transaction type there is a corresponding function which controls transaction activity for the remainder of its time in the system. These functions do not have symbollic names but are designated 1, 2, 3,---etc. and constitute the curricula to be followed by the students. The transaction type number and the curriculum function number are equal.

5. The numbers contained in the curriculum functions are learning module numbers and are used as arguments for reference to other functions to determine module type code, completion time, etc.

6. Module type code is discussed under the model generation subsystem and determines the support requirements for each module.

FUNCTION NAME	TYPE	DESCRIPTION
1,2,3, etc.	LIST	Curriculum Functions relate step number to module number.
MODT	LIST	Relates module number to module type code.
TYME	LIST	Relates module number to completion time.
SPRD	LIST	Relates module number to completion time variation.
INSTT	LIST	Relates instructor group number to module number.
INSTN	LIST	Relates instructor group size to module number.
STUDT	DISCRETE	Relates student type to random number distribution.
FACT	LIST	Relates facilities groups to module number.
TEAM	LIST	Relates team size to module number.
REMM	LIST	Relates remedial module groups to module numbers.
UNSHL	LIST	Relates staff groups to module numbers.

SYSTEM GENERATED GPSS FUNCTIONS

FIGURE III-8

7. Once the student begins a module, resources are obtained in the following order:

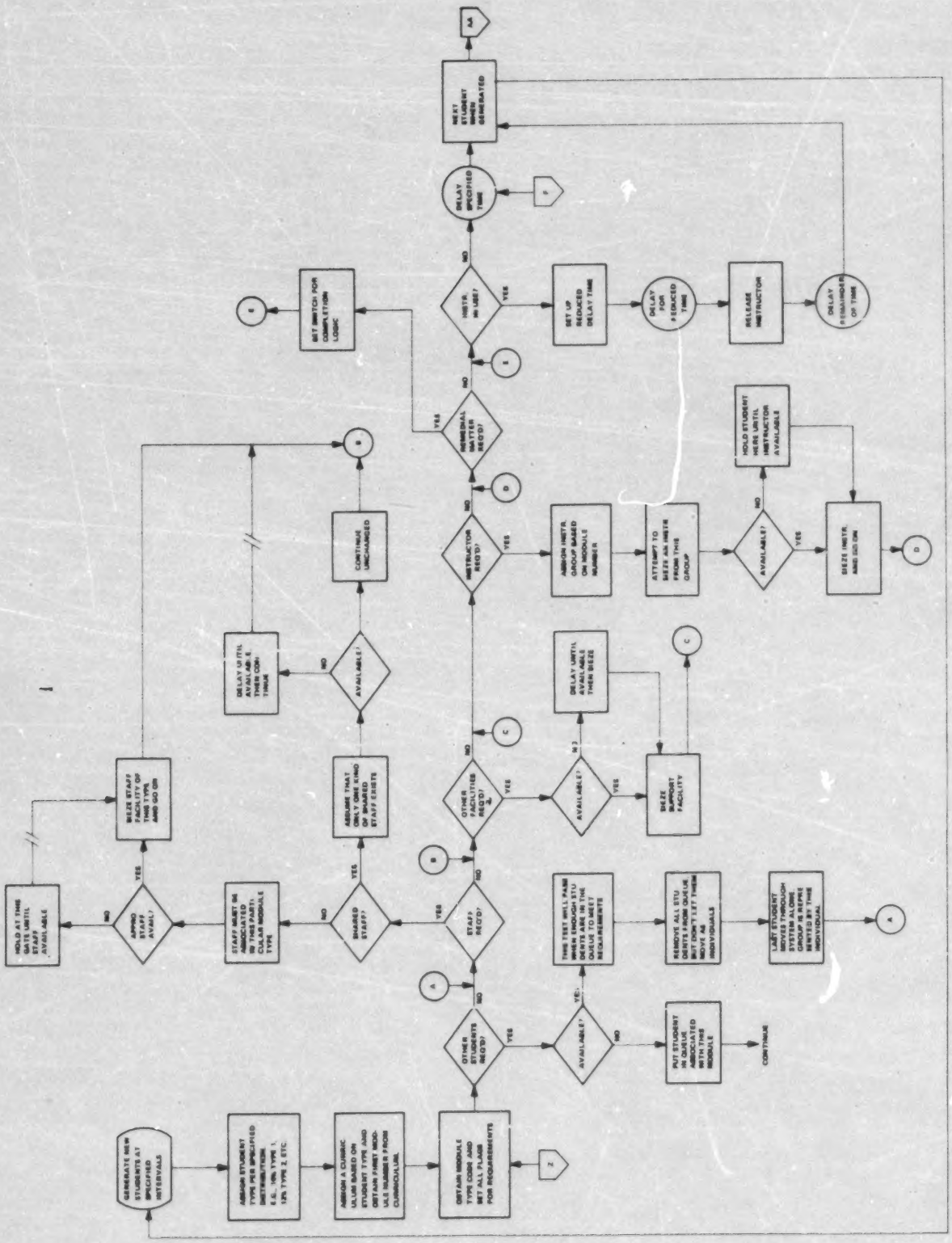
   a) Instructional materials - STORAGES 1-50
   b) Staff personnel - STORAGES 101 - 125
   c) Other facilities - STORAGES 51 - 100
   d) Instructor - FACILITIES 1 - 50

   As the flow diagram shows, the transaction (student) always waits when a particular resource is needed but not available.

8. The students move through each module (as controlled by the curriculum function) until the last module is completed (this condition is noted when the next curriculum step = 50). The student is then terminated and the student head count and graduation statistics updated.

SIMULATION OUTPUT. All output is generated via the GPSS report generator. No standard GPSS output appears (although it is easily obtained by editing an OUTPUT card into DUMMY GPSS).

Two cards: UNLIST and SKIP are used to suppress output of any assembly listings. These listings are suppressed in order to keep the output files as small as possible.



GPSS MODEL LOGIC FLOW
FIGURE III-9

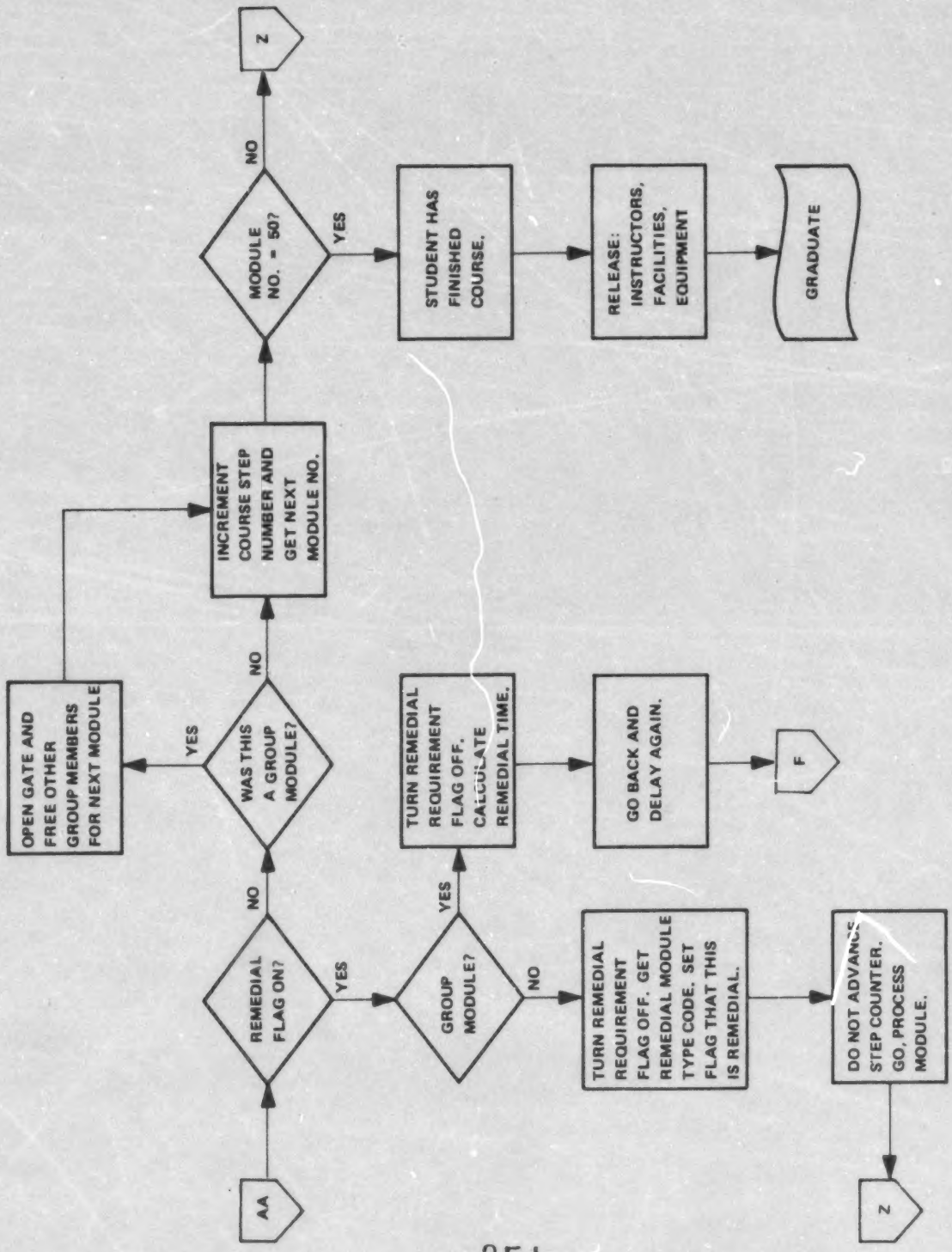


FIGURE III- 9 (CONTINUED)

```
*  EDUCATIONAL TECHNOLOGY EVALUATION MODEL  LEVEL 1
       SIMULATE
       UNLIST
       SKIP          ,,1
 ONE   VARIABLE      FN$TYME/2    THESE ARE USED TO COMPUTE MEAN
 TWO   VARIABLE      FN$SPRD/2    AND SPREAD FOR TEMEDIAL MODULES
 THREE VARIABLE      X*P2/X*P1    COMPUTE AVG TRANSIT TIME
 FOUR  FVARIABLE     (1-FN$INTIM)*FN$TYME
*  FOUR COMPUTES REDUCED TIME FOR INSTRUCTOR IN USE
 FIVE  FVARIABLE     (1-FN$INTIM)*FN$SPRD
*  FIVE COMPUTES REDUCED SPREAD FOR INSTRUCTOR IN USE
 SIX   VARIABLE      FN$TYME-P15
 SEVN  VARIABLE      FN$SPRD-P16
 ATE   VARIABLE      P15/2        VARIABLE TO COMPUTE REDUCED
 NINE  VARIABLE      P16/2              TIMES FOR REMEDIAL MODULES.
*****************************************************************
*****************************************************************
*****************************************************************
***
***                  THE FUNCTION CARDS WITHIN
***
***                  THIS AREA ARE SAMPLE DATA
***
***                  ONLY AND SHOULD
***
***                  BE REPLACED BY THE USER-GENERATED TABLES
***
***                  DESCRIBING THE CURRICULA, INSTRUCTORS, AND
***
***                  EQUIPMENT, ETC. NECESSARY TO DEFINE THE
***
***                  ILS TO BE SIMULATED
***
***
 INTIM FUNCTION      RN5,C3
Ø      1.Ø    .2     1.Ø    1.Ø    .9
*  START
*  INTIM ASSIGNS FRACTIONAL TIMES WHEN INSTRUCTOR IN USE
*****************************************************************
*****************************************************************
*****************************************************************
       GENERATE      4,Ø,,,,2Ø
       SAVEVALUE     1Ø1+,K1          INCREMENT NO. OF STUDENTS
       TEST G        X1Ø1,V$AOB,OKK   TEST FOR AOB EXCEEDED
       SAVEVALUE     1Ø1-,K1
       TERMINATE
 OKK   ASSIGN        9,1              OK, GO AHEAD, SET P9 = 1
       ASSIGN        1,FN$STUDT          ASSIGN STUDENT TYPE TO P1
       ASSIGN        2,FN*1              GET FIRST MODULE NUMBER IN P2
       ASSIGN        3,FN$MODT           PUT MODULE TYPE IN P3
 ZZZ   SAVEVALUE     51,P3            MOVE MODULE TYPE TO X51
       TEST GE       X51,K16,BBB      TEST MOD TYPE GE 16
       ASSIGN        8,1                 YES SET LOGIC SWITCH FOR REMEDIAL
       SAVEVALUE     51-,K16          REDUCE MODULE DESIGNATOR BY 16
 BBB   TEST GE       X51,K8,CCC       TEST MODULE TYPE GE 8
       ASSIGN        7,1              SET FACILITIES REQUIRED SWITCH
       SAVEVALUE     51-,K8           REDUCE MODULE DESIGNATOR BY 8
 CCC   TEST GE       X51,K4,DDD       TEST MODULE TYPE GE 4
       ASSIGN        6,1              SET STAFF REQUIRED SWITCH
```

DUMMY GPSS LISTING
TABLE III-6

```
        SAVEVALUE  51-,K4        REDUCE MODULE TYPE BY 4
 DDD    TEST GE    X51,K2,EEE     TEST GE 2
        ASSIGN     5,1           SET OTHER STUDENTS REQUIRED
        SAVEVALUE  51-,K2        REDUCE MODULE TYPE BY 2
 EEL    TEST GE    X51,K1,FFF    TEST GE 1
        ASSIGN     4,1           EQUALS 1-SET INSTRUCTOR REQUIRED
*                                ALL REQUIRED SWITCHES ARE SET
*                 SWITCH ASSIGNMENTS ARE AS FOLLOWS
*                 PARAMETER 1 - STUDENT TYPE
*                 PARAMETER 2 - CURRENT MODULE NUMBER (FOR THIS XACT)
*                 PARAMETER 3 - CURRENT MODULE TYPE
*                 PARAMETER 4-8 - SWITCHES FOR INSTRUCTOR NEEDED ETC
*                 PARAMETER 4 - INSTRUCTOR NEEDED
*                 PARAMETER 5 - GROUP MODULE
*                 PARAMETER 6 - STAFF REQUIRED
*                 PARAMETER 7 - FACILITIES REQUIRED
*                 PARAMETER 8 - REMEDIAL MATTER OUTSIDE MODULE
*                 PARAMETER 9 - CURRENT MODULE STEP (FOR THIS XACT)
*                 PARAMETER 1Ø -FLAG THAT SAYS REMEDIATION IS NEEDED
*                 PARAMETER 11 -FLAG THAT DESIGNATES CURRENT MODULE
*                   IS A REMEDIAL MODULE
 FFF    TEST E     K1,P5,GRUP    TEST FOR OTHER STUDENTS REQUIRED
        ASSIGN     12,FN$TEAM    PUT NUMBER OF TEAM MEMBERS IN 12
        ASSIGN     2+,1ØØ        ASSIGN QUEUE NUMVER = TYPE + 1ØØ (GROUP)
        QUEUE      *2            YES THEY ARE-TEST FOR OTHERS
        ASSIGN     1Ø,P2         P1Ø WILL HAVE P2+8  AS LOGIC SWITCH NO.
        ASSIGN     1Ø+,8
        TEST E     Q*1Ø,Ø,FIX    IF Q*1Ø IS EMPTY LET FIRST
        LOGIC R    *1Ø           TEAM MEMBER GO THROUGH
 FIX    TEST GE    Q*2,P12,HLD    TEST FOR REQ. NO. IN QUEUE
        LOGIC S    *2            YES THERE ARE, OPEN GATE FOR OTHERS
 HLD    GATE LS    *2            HOLD HERE UNTIL TEAM COMPLETE
        DEPART     *2            REMOVE FROM TEAM QUEUE
        GATE LR    *1Ø,ENDS      DIVERT IF OTHER THAN FIRST MEMBER
        LOGIC S    *1Ø           FIRST MEMBER SETS GATE TO DIVERT
        ASSIGN     2-,1ØØ        SET BACK TO STUDENT TYPE
        BUFFER
 GRUP   TEST E     K1,P11,GRUPE   TEST FOR REMEDIAL MODULE
 GROPE  ASSIGN     2+,1ØØ        MOVE POINTER TO REMEDIAL MODULE STORAGE
        ENTER      *2            GET REMEDIAL MODULE
        ASSIGN     2-,1ØØ        RESET POINTER
        TRANSFER   ,COM1         YOU MAY NOW PROCEED
 GRUPE  ENTER      *2            POINTER FOR MODULE STORAGE IS P2
 COM1   TEST E     K1,P6,NINS     TEST FOR STAFF NEEDED
 NSH    ASSIGN     2Ø,FN$UNSHL      PUT FACILITY LOCATION AND
        ASSIGN     17,35         LOGIC SWITCH 35 FOR THIS FACILITY
        ASSIGN     2Ø+,125           ASSIGN CORRECT QUEUE AND STORAGE
        QUEUE      *2Ø
        ENTER      *2Ø               GET UNSHARED STAFF
        DEPART     *2Ø               LEAVE QUEUE WHEN STAFF OBTAINED
        TRANSFER   ,NINS
 SHR    GATE LR    43            SHARED STAFF-AVAILABLE IF RESET
 MMM    TEST E     K1,P4,NFAC    TEST FOR INSTRUCTOR REQUIRED
        ASSIGN     14,FN$INSTT       PUT QUEUE
        QUEUE      *14           QUEUE PRIOR TO TRYING FOR INSTRUCTOR
 LLL    ASSIGN     14,FN$INSTT       PUT HIGHEST INST. FAC LOC IN P14
        ASSIGN     12,1,INSTN        PUT NUMBER OF INSTRUCTORS IN 12
        ASSIGN     17,36         LOGIC SWITCH 36 FOR INSTRUCTORS
 LLL1   TRANSFER   SBR,CCCS,18PH ATTEMPT TO SEIZE INSTRUCTOR
        GATE LR    *17           HOLD HERE UNTIL SOMEBODY RELEASES
        TRANSFER   ,LLL          AN INSTRUCTOR,THEN TRY AGAIN
        ASSIGN     19,P14        MOVE INSTRUCTOR POINTER TO P19
        ASSIGN     14,FN$INSTT    RESTORE QUEUE NO. IN P14
```

TABLE III-6 (CONTINUED)

```
        DEPART      *14          RELEASE AFTER GETTING INSTRUCTOR
        ASSIGN      14-,12Ø      RESET P14 TO CORRECT VALUE
        ASSIGN      12,FN$FACT
        ASSIGN      12+,5Ø             RESET 12 TO OFAC
        TRANSFER    ,NFAC
NINS    TEST E      K1,P7,MMM    TEST FOR OTHER FACILITIES
        ASSIGN      12,FN$FACT   PUT FACILITY STORAGE NUMBER IN 12
        ASSIGN      12+,5Ø       MOVE POINTER TO CORRECT QUEUE
        QUEUE       *12          QUEUE PRIOR TO ENTERING STORAGE
        ENTER       *12          OCCUPY OTHER FACILITIES IF POSSIBLE
        DEPART      *12          LEAVE QUEUE WHEN FINISHED
        TRANSFER    ,MMM
NFAC    TEST E      K1,P8,DEL    TEST FOR REMEDIATION SELF CONTAINED
        TRANSFER    .6,REM,DEL   NOT SELF CONTAINED. SPLIT 4Ø PERCENT
REM     ASSIGN      8,2          SET REMEDIAL NEEDED
        TRANSFER    ,DEL         GO DELAY
ENDS    QUEUE       *1Ø
        TEST E      Q*2,Ø,STT    TEST FOR ALL REMOVED
        LOGIC R     *2           CLOSE GATE TO Q*2 EXIT
STT     GATE LR     *1Ø          HOLD HERE FOR REST OF THEM
        DEPART      *1Ø
        ASSIGN      2-,1ØØ       SET BACK TO STUDENT TYPE
        TEST E      Q*1Ø,Ø,NNFC  KEEP SENDING XACTS TO COMPLETION
        LOGIC S     *1Ø          UNTIL QUEUE IS EMPTY. THEN CLOSE
        TRANSFER    ,NNFC        GATE.
GRAD    ASSIGN      2,P1         MOVE POINTER TO P2
        ASSIGN      2+,25        SET IT 1Ø-15
        SAVEVALUE   P1+,1        ADD 1 TO GRAD COUNT
        SAVEVALUE   P2+,M1       ADD TRANSIT TIME TO TOTAL
        SAVEVALUE   P1,V$THREE,XH  COMPUTE AVG TIME AND PUT IN P3 SAVE
        SAVEVALUE   1Ø1-,K1      REDUCE AOB FOR GRADUATION
TRM1    TERMINATE   1
NRM     TEST E      K1,P11,NRM1        TEST FOR REMEDIAL MODULE
        ASSIGN      15,V$ONE           YES, SET UP REDUCED COMPLETION
        ASSIGN      16,V$TWO           TIME AND SPREAD
        TRANSFER    ,DELS              GO DELAY
NRM1    ASSIGN      15,FN$TYME   FN 5 HAS MODULE TIME
        ASSIGN      16,FN$SPRD   FN 6 HAS SPREAD FOR MODULE
        TRANSFER    ,DELS        GO TO DELAY FOR THIS MODULE
DEL     TEST E      K1,P4,NRM    TEST FOR INSTRUCTOR NEEDED
        ASSIGN      15,V$FOUR    ASSIGN REDUCED COMPLETION TIME
        ASSIGN      16,V$FIVE     AND SPREAD.
        TEST E      K1,P11,JDEL        TEST FOR REMEDIAL
        ASSIGN      15,V$ATE           IT ARE ONE, REDUCE
        ASSIGN      16,V$NINE          COMPLETION TIME
JDEL    ADVANCE     P15,P16            DELAY FOR THIS MODULE
        RELEASE     *19          LET THE INSTRUCTRO GO
        LOGIC R     *17
        BUFFER
        ASSIGN      15,V$SIX     SET UP REMAINDER
        ASSIGN      16,V$SEVN    OF DELAY.
        ADVANCE     P15,P16      DELAY REST OF TIME
        TRANSFER    ,DELP        GO FNISH UP
DELS    ADVANCE     P15,P16      DELAY FOR THIS MODULE
*   AT THIS POINT THE STUDENT HAS BEEN SET UP TO DO THE MODULE
*   THE COMPLETION LOGIC TAKES CARE OF THING LIKE REMEDIAL
*   MODULES,LAST MODULE ETC.
DELP    TEST E      K2,P8,ABD    TEST FOR REMEDIAL MODULE NEEDED
        TEST E      K1,P5,ABM    TEST FOR GROUP XACT
        ASSIGN      8,Ø          TURN OFF REMEDIAL FLAG
        ASSIGN      11,Ø         TURN OFF FLAG FOR REMEDIAL MODULE
```

TABLE III-6 (CONTINUED)

```
        ASSIGN      15,V$ONE     AND SPREAD. FUNCTION 3 IS REMEDIAL TIME
        ASSIGN      16,V$TWO     FN4 IS REMEDIAL SPREAD
        TRANSFER    ,DELS        YES, GO DELAY FULL TIME
ABM     LEAVE       *2               GIVE UP MODULE
        TEST E      K1,P7,NNF1
        ASSIGN      12,FN$FACT
        ASSIGN      12+,5Ø            SET TO OTHER FACILITIES
        LEAVE       *12
NNF1    ASSIGN      11,1             SET FLAG THAT THIS IS A REMEDIAL MODUL
        BUFFER
        ASSIGN      3,FN$MODR    SET REMEDIAL MODULE TYPE
        ASSIGN      4,Ø
        ASSIGN      5,Ø
        ASSIGN      6,Ø
        ASSIGN      7,Ø
        ASSIGN      8,Ø
        TRANSFER    ,ZZZ         PROCESS LIKE REGULAR MODULE.
ABD     TEST E      K1,P11,AAB   TEST FOR COMPLETION OF REMEDIAL MOD
        ASSIGN      11,Ø         YES IT WAS,TURN OFF REMEDIAL FLAG
        TEST L      P2,K1ØØ,REL
        ASSIGN      2+,1ØØ           SET TO REM MOD POINTER
AAB     TEST E      K1,P5,AABB   TEST FOR GROUP XACT FINISHED
        LOGIC R     *1Ø          OPEN GATE FOR REST OF TEAM
        BUFFER                   LET OTHER TEAM MEMBERS MOVE
AABB    TRANSFER    ,REL
REL     LEAVE       *2           GIVE UP MODULE
        TEST NE     Ø,P2Ø,DDD1   TEST FOR UNSHARED STAFF
        LEAVE       *2Ø          RELEASE STAFF
        ASSIGN      2Ø,Ø         TURN OFF FLAG AFTER RELEASING
DDD1    LOGIC R     34           OPEN GATES FOR MODULE, INSTRUCTOR
        LOGIC R     35           OR STAFF AGAIN
        LOGIC R     36
        TEST E      K1,P7,NNFC   TEST FOR OTHER FACILITIES
        LEAVE       *12          RELEASE OTHER FACILITIES
        BUFFER
NNFC    ASSIGN      4,Ø          ZERO OUT ALL LOGIC SETTINGS
        ASSIGN      5,Ø          ZERO OUT ALL LOGIC STTINGS
        ASSIGN      6,Ø
        ASSIGN      7,Ø
        ASSIGN      8,Ø
        ASSIGN      11,Ø         ZERO REMEDIAL SWITCH
        TRANSFER    ,AAC         GO TO NEXT MODULE
AAC     ASSIGN      9+,1         STEP P9 TO NEXT COURSE STEP
        ASSIGN      2,FN*1       PUT NEW MOD NO. IN P2
        TEST NE     5Ø,P2,GRAD
        ASSIGN      3,FN$MODT    ASSIGN NEW MODULE TYPE
        TRANSFER    ,ZZZ         GO PROCESS NEXT MODULE
*  THIS IS A SUBROUTINE FOR SEIZING MODULES,INSTRUCTORS OR WHAT EVER
*       ENTER WITH NUMBER OF ENTITIES TO BE TESTED IN P12
*                  NUMBER (LOCATION) OF HIGHEST ENTITY IN P14
*                  LOCATION OF LOGIC SWITCH IN P17
*       RETURNS WITH SEIZED ENTITY IN P14. IF FACILITY UNAVAILABLE
*       CLOSES LOGIC GATE AT RETURN POINT
*       RETURNS TO TRANSFER+1 WHEN BAD,TRANSFER+2 WHEN GOOD
CCCS    GATE U      *14,SEZ      CHECK FACILITY IN USE
        ASSIGN      14-,1        YES IT IS,TRY NEXT
        LOOP        12,CCCS
        LOGIC S     *17          DIDNT FIND ANY CLOSE GATE
        TRANSFER    PH,18,1      RETURN TO CALLER
SEZ     SEIZE       *14          FACILITY AVAILABLE SEIZE IT
        TRANSFER    PH,18,3      DIFFERENT RETURN WHEN SUCCESSFUL
        START       5Ø
```

TABLE III-6 (CONTINUED)

```
        REPORT
        SPACE       25
*                                STUDENT STATISTICS
        SPACE       2
*            STUDENT TYPE   NUMBER COMPLETING   AVG. COMPLETION TIME
        FORMAT      1-25/X1,X2,XH2
        SPACE       2
        EJECT
*                                INSTRUCTOR STATISTICS
        SPACE       1
*      INSTRUCTOR NO.   PERCENT UTIL.   AVG. WAIT TIME  MAX. NO. WAITING
        FORMAT      1-5Ø/F1,F2,Q8,Q2
        SPACE
*                                MODULE UTILIZATION
        SPACE       1
*           MODULE NO.       PERCENT USAGE   NO.USING      AVG. TIME
        SPACE       1
 2      FORMAT      1-3Ø/S1,S4,S5,S6
        SPACE
        EJECT
*                     OTHER FACILITIES UTILIZATION
        SPACE       1
*       FACILITY NUMBER    PERCENT USAGE   NO. USING      AVG. TIME
        SPACE       1
 2      FORMAT      51-7Ø/S1,S4,S5,S6
        SPACE
        EJECT
*                                QUEUE DATA
        SPACE       1
*       QUEUE NUMBER       MAX CONTENTS   AVG WAIT TIME      AVG CONTENTS
        FORMAT      1-35/Q1,Q2,Q7,Q3
        FORMAT      51-7Ø/Q1,Q2,Q7,Q3
        FORMAT      1Ø1-15Ø/Q1,Q2,Q7,Q3
        EJECT
*      STAFF GROUP       PERCENT UTIL.   AVG WAIT TIME   MAX NO WAITING
        FORMAT      125-14Ø/S1,S4,Q8,Q2
*FINISH
        END
/*

Ø7.47.14
```

TABLE III-6 (CONTINUED)

SECTION IV.

SYSTEM CAPABILITIES/REQUIREMENTS AND RESOURCES MODEL

MODEL DESCRIPTION

The SCRR model is a set of sixteen (16) VP/CSS Executive procedures, eight (8) FORTRAN programs, two (2) PL/1 programs, and one (1) MPS III/OL procedure. All of these procedures and programs are executed under the control of the seven (7) primary VP/CSS Executives depicted in Figure IV-1. Figure IV-1 shows the overall flow through the seven executives and the connection with the RAMIS scratch data base procedures. The various input and output data files and all executives, programs and procedures executed by the primary executives do not appear, since some are accessed several times and the overview flowchart would be nearly unintelligible.

Table IV-1 gives a brief description of the VP/CSS Executive programs utilized by the SCRR model. All RAMIS procedures, FORTRAN programs, PL/1 programs, and the MPS III/OL procedure are executed by these executives. This section of the Programmer's Manual has been organized based on these VP/CSS executive programs. For example, the RAMSCRR subsection contains the verbal descriptions, flowcharts, and listings for the RAMSCRR EXEC and the five RAMIS procedures executed by RAMSCRR. The SCRR1 subsection contains descriptions, flows, and listings for the SCRR1 EXEC, the MPS III/OL procedure, and the FORTRAN subroutine executed under control of MPS III.

Table IV-2 gives a list of the input and output data files used by each of the executive programs. All files are stored on disk and are named by a unique three-part identifier composed of a filename, a filetype and a filemode. Only the filename and filetype are shown in Table IV-2. All these files have a filemode of P which indicates that they reside on the P (Permanent) disk.

OVERVIEW OF SCRR EXECUTIVE PROGRAMS

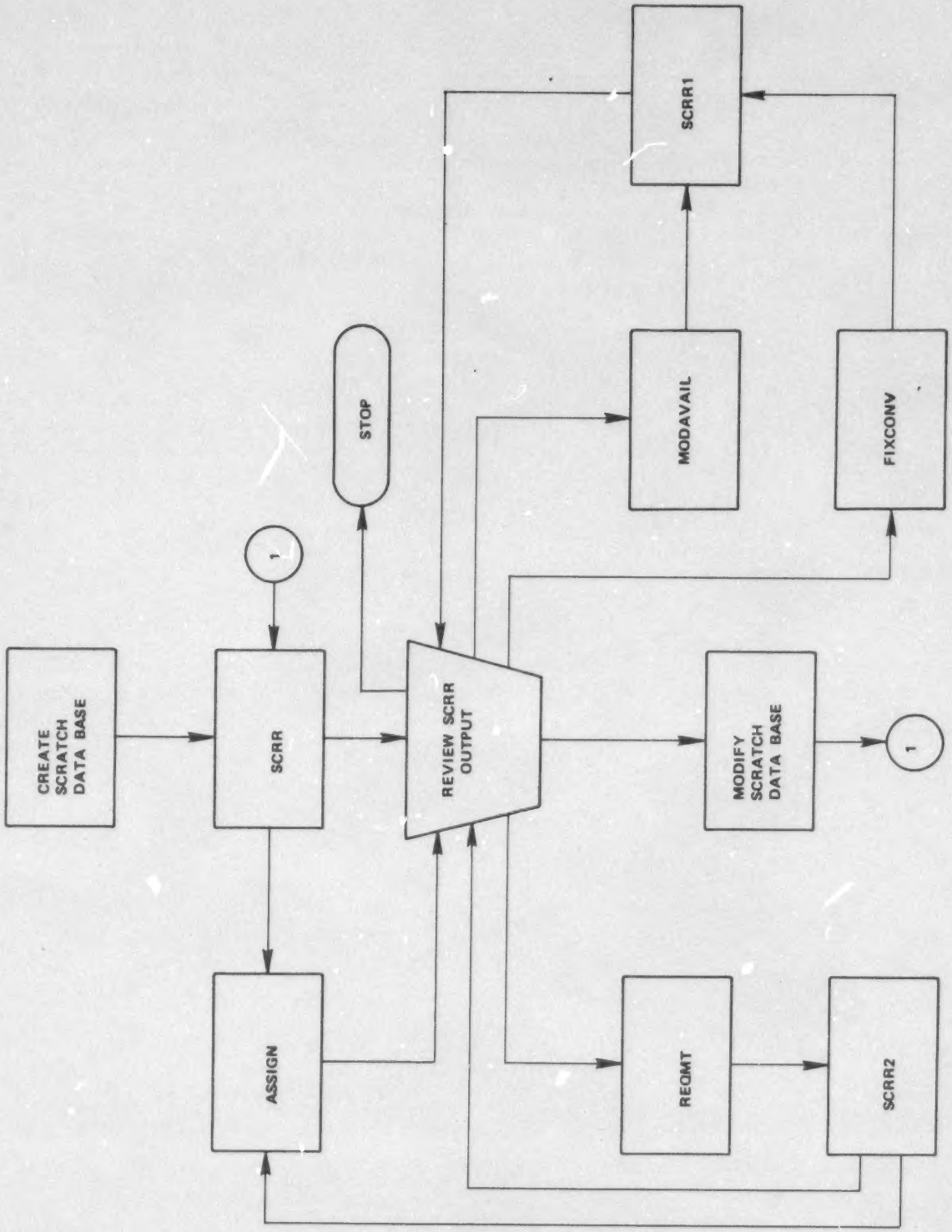


FIGURE IV-1

SCRR EXECUTIVE PROGRAMS DESCRIPTION

EXEC NAME	DESCRIPTION
SCRR	A top level executive which executes the following programs: RAMSCRR, BLANK, IF1, IF2, SCRRSORT, IF3, IF4, COMB, and SCRR1.
RAMSCRR	Uses the RAMIS data base procedures SCRR INST, SCRR FACIL, SCRR IAVAIL, SCRR FAVAIL, and SCRR CRSE to retrieve all SCRR model basic input data from the three scratch data files MODCOURSE, MODINST, and MODFACIL and to derive additional data from these basic inputs.
BLANK	Deletes all internal blank characters from facility names in the files FILE FACIL and FILE FAVAIL.
IF1	Formats NAME and ROW data for LP input.
IF2	Formats LP COLUMN data including instructor and facility requirements and objective function coefficients.
SCRRSORT	Sorts IF2 output by course CDP number.
IF3	Formats LP RHS data section, including instructor and facility availabilities.
IF4	Formats BOUNDS data section, including course convenings limits, of LP input data.
COMB	Combines outputs from IF1, IF2, IF3, and IF4 into a single file MATRIX1 DATA which is directly readable through MPS III/OL Terminal Control Language.
SCRR1	Establishes all control data (i.e., input, output, and internal processing files) required by MPS III/OL system and initiates the LP processing.
SCRR2	A top level executive which executes the following programs: BLANK, IF2, SCRRSORT, COMB, SCRR1.
ASSIGN	Compiles and prints a list of instructor assignments and contact hour requirements.
MODAVAIL	Modifies resource availability data prior to LP optimization.
REQMT	Permits user to specify percentage changes to variables which impact instructor requirements.
FIXCONV	Modifies the LP input data to fix the number of course convenings at the current level.

TABLE IV-1

SCRR DATA FILE SUMMARY

EXEC	INPUT FILES	OUTPUT FILES
RAMSCRR	RAMIS files MODCOURSE, MODINST, MODFACIL	FILE INST, FILE IAVAIL, FILE FACIL, FILE FAVAIL, FILE CRSE
BLANK	FILE FACIL, FILE FAVAIL	FILE FACIL, FILE FAVAIL
IF1	FILE IAVAIL, FILE FAVAIL	FILE IF1
IF2	FILE INST, FILE FACIL, FILE CRSE	FILE IF2
SCRRSORT	FILE IF2	FILE IF2
IF3	FILE IAVAIL, FILE FAVAIL	FILE IF3
IF4	FILE CRSE	FILE IF4
COMB	FILE IF1, FILE IF2, FILE IF3, FILE IF4	MATRIX1 DATA
SCRR1	MATRIX1 DATA, SCRR RUN, SCRRFORM LNKLIB	SCRR OUTPUT
ASSIGN	FILE INST, FILE CRSE, FILE IAVAIL	ASSIGN LISTING
MODAVAIL	MATRIX1 DATA	MATRIX1 DATA
REQMT	RAMIS files MODCOURSE and MODFACIL	FILE FACIL, FILE INST, FILE CRSE
FIXCONV	MATRIX1 DATA	MATRIX1 DATA
MAXCONV	MATRIX1 DATA	MATRIX1 DATA

TABLE IV-2

SCRR EXEC DESCRIPTION

The purpose of executive SCRR is to execute all SCRR model-related executives through a single keyboard entry.

Upon entry, a test is made to determine if core size is equal to or greater than 320K. If core size is less than 320K, a message directing the user to SET CORE 320 is issued and SCRR EXEC is terminated. If the core requirement is met, a message describing the objective of the EXEC is printed and the user is given the alternative of proceeding with model execution or terminating the SCRR EXEC at this point. If the user elects to proceed with SCRR model execution, the following EXECs are executed serially:

RAMSCRR
BLANK
IF1
IF2
SCRRSORT
IF3
IF4
COMB
SCRR1

Each of these executives is described in detail on the following pages.

SCRR EXEC FLOWCHART

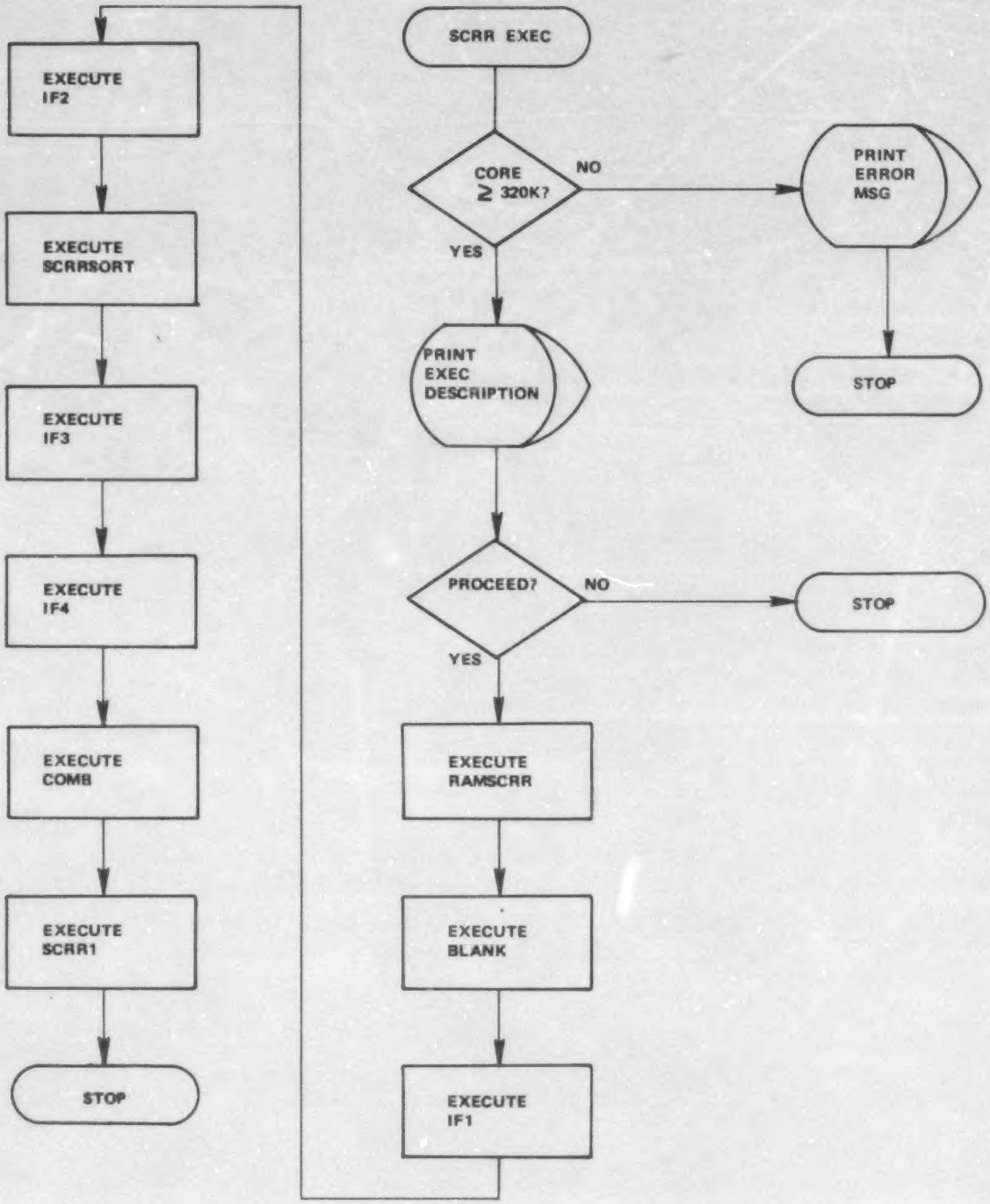


FIGURE IV-2

SCRR PROGRAM LISTING

```
SCRR      EXEC
         VP/CSS

&IF &CORE GE 320 &SKIP 3
&PRI REQUIRED CORE IS 320K. PLEASE TYPE: SET CORE 320
&PRI THEN REISSUE COMMAND: SCRR
&EXIT
&PRI THIS PROGRAM PERFORMS THE LINEAR PROGRAMMING OPTIMIZATION
&PRI OF THE RESOURCE DATA CURRENTLY STORED IN THE SCRATCH DATA FILES.
&PRI NO FURTHER USER INTERVENTION WILL BE REQUIRED.
&PRI DO YOU WISH TO PROCEED? (YES/NO)
&READ ARGS
&IF &1 NE YES &GOTO -EXIT
&STACK KT
EXEC RAMSCRR
EXEC BLANK
EXEC IF1
EXEC IF2
EXEC SCRRSORT
&STACK KT
EXEC IF3
EXEC IF4
EXEC COMB
EXEC SCRR1
-EXIT &EXIT
```

TABLE IV-3

RAMSCRR EXEC DESCRIPTION

All interface between the SCRR model and the DOTS data base is performed by the RAMSCRR EXEC. RAMSCRR utilizes a series of RAMIS data base procedures to retrieve (and to calculate when required) instructor and facility requirements and availabilities and course parameters from the three RAMIS data files MODINST, MODCOURSE, and MODFACIL.

Initial RAMSCRR statements suppress all terminal output and gain access to RAMIS data base software. Then the output file FILE INST is established for the RAMIS procedure SCRR INST, and SCRR INST is executed. Similarly, output files FILE FACIL, FILE FAVAIL, FILE IAVAIL, and FILE CRSE are defined and RAMIS procedures SCRR FACIL, SCRR FAVAIL, SCRR IAVAIL, and SCRR CRSE are executed. RAMSCRR EXEC concludes by removing the RAMIS software disk from the user's virtual machine.

SCRR INST DESCRIPTION. The purpose of the RAMIS procedure SCRR INST is to calculate instructor requirements in hours by course and by instructor. The basic data required to determine instructor requirements are stored in the RAMIS course file MODCOURSE.

SCRR INST first calculates the total number of contact hours required (TOT&HRS) for each student/instructor ratio (SIR) within each course (CDP). The class capacity is multiplied by the historical course utilization percentage to determine average class size. Average class size is divided by the SIR to determine the number of instructors (NO&INST) required at each SIR. NO&INST is multiplied by the required contact hours at each SIR to arrive at TOT&HRS.

The number of instructors physically assigned at each SIR is determined by summing the value of ASN (1 = assigned, 0 = not assigned). TOT&HRS is then divided by the sum of ASN to determine the number of hours required per instructor per SIR (INST&HRS). Summing INST&HRS by course (CDP) and by instructor is the final step of the SCRR INST procedure. Output is written on FILE INST. The record format for FILE INST is shown below:

FIELD NAME	CDP NO	INST NAME	REQ'D HRS
TYPE	A	A	F
LENGTH	4	16	4.1
RELATIVE POS	1-4	5-20	21-25

SCRR FACIL DESCRIPTION. The purpose of the RAMIS procedures SCRR FACIL is to determine facility requirements in hours by course and by facility. These data are stored in the RAMIS facility file MODFACIL.

SCRR FACIL retrieves the required hours from MODFACIL by course and by building and room number. Output is written on FILE FACIL. The record format for FILE FACIL is shown below:

FIELD NAME	CDP NO.	BLDG.	RM	REQ'D HRS.
TYPE	A	A	A	F
LENGTH	4	5	3	5.1
RELATIVE POS.	1-4	5-9	10-12	13-18

SCRR FAVAIL DESCRIPTION. The purpose of the RAMIS procedure SCRR FAVAIL is to determine facility availability. Annual availability is stored in the RAMIS facility file MODFACIL by building and room.

A building and room combination may appear in the MODFACIL file more than once. Annual available hours is included on each MODFACIL record. SCRR FAVAIL retrieves the available hours associated with only the first appearance of each facility in the file. Output is written on FILE FAVAIL. The record format for FILE FAVAIL is shown below:

FIELD NAME	BLDG.	RM	AVAIL HRS.
TYPE	A	A	I
LENGTH	5	3	4
RELATIVE POS.	1-5	6-8	9-12

SCRR IAVAIL DESCRIPTION. The purpose of the RAMIS procedure SCRR IAVAIL is to determine instructor availability for classroom instruction. The basic information from which availability is calculated is stored in the RAMIS instructor file MODINST.

Actual classroom instruction time by course is recorded and entered into the DOTS data base on a weekly bases. In addition to the weekly totals, quarter-to-date and year-to-date totals are maintained. SCRR IAVAIL computes the number of days between the first data entry (the first day of the current fiscal year) and the last data entry, and divides this number into 365. The resulting factor (FAC) is multiplied by the year-to-date total classroom instruction hours to project annual classroom instruction time for each course. If no classroom instruction time has been recorded to date by an instructor, a default value of one (1) hour is entered. SCRR IAVAIL sums the projected annual instruction time for all courses by instructor. Output is written on FILE IAVAIL. The record format for FILE IAVAIL is shown below:

FIELD NAME	INST NAME	AVAIL HRS.
TYPE	A	I
LENGTH	16	4
RELATIVE POS.	1-16	17-20

SCRR CRSE DESCRIPTION. The purpose of the RAMIS procedure SCRR CRSE is to record the convening frequency and average class size for each course. The required data are stored in the RAMIS course file MODCOURSE.

SCRR CRSE first calculates average class size by multiplying class capacity by percent utilization. Convening frequency and class size are recorded by course. Output is written on FILE CRSE. The record format for FILE CRSE is shown below:

FIELD NAME	CDP NO'.	CLASS SIZE	CONV
TYPE	A	I	I
LENGTH	4	3	I
RELATIVE POS.	1-4	5-7	8-10

RAMSCRR EXEC FLOWCHART

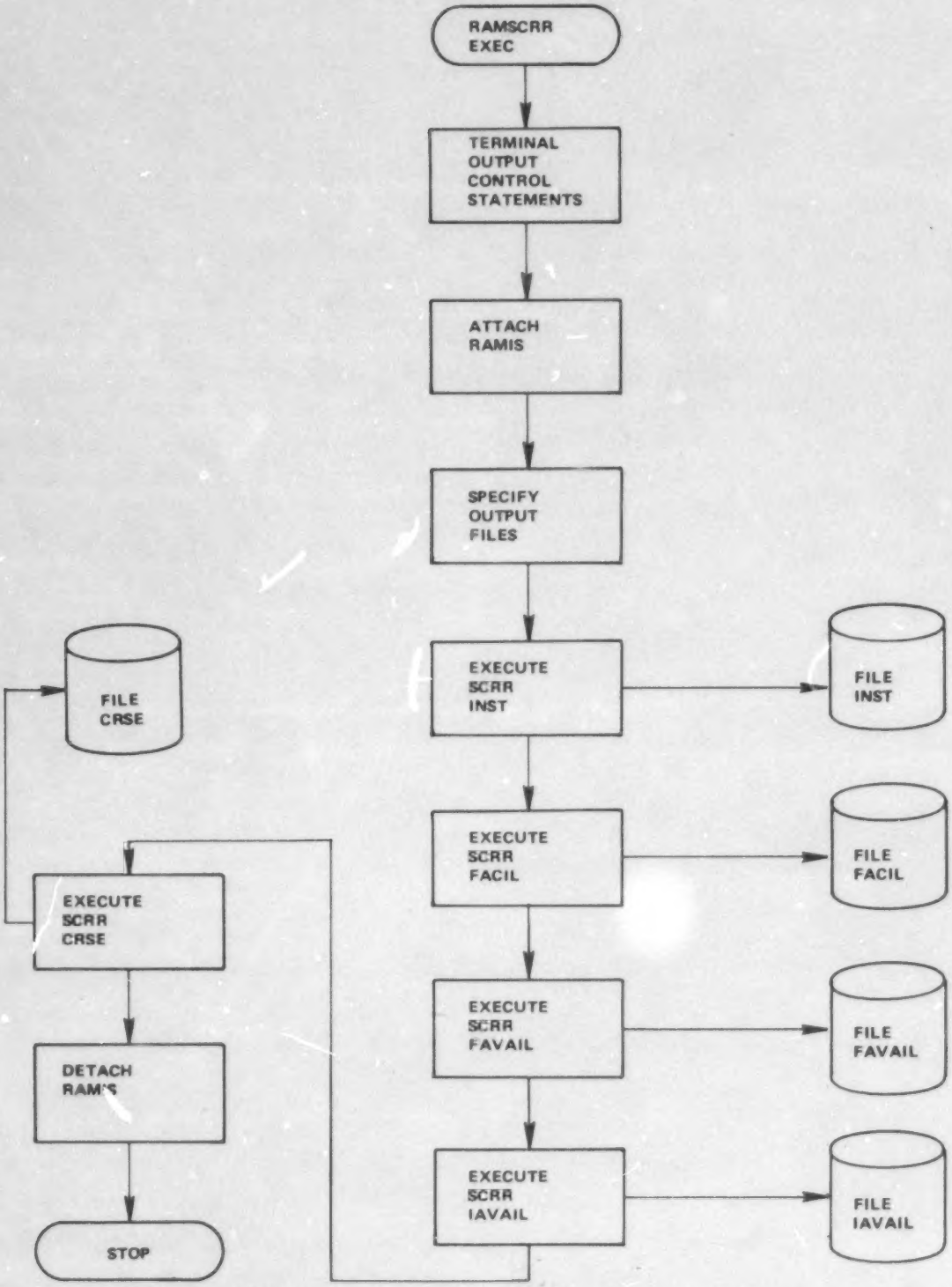


FIGURE IV-3

RAMSCRR PROGRAM LISTINGS

```
RAMSCRR  EXEC
         VP/CSS


&TYPE OFF
&STACK KT
FILEDEF 4 DSK FILE INST RE F LR 25
ATTACH RAMIS ALL BRIEF
EXEC RAMISIN SCRR INST
FILEDEF 4 DSK FILE IAVAIL RE F LR 20
EXEC RAMISIN SCRR IAVAIL
FILEDEF 4 DSK FILE CRSE RE F LR 10
EXEC RAMISIN SCRR CRSE
FILEDEF 4 DSK FILE FACIL RE F LR 22
EXEC RAMISIN SCRR FACIL
FILEDEF 4 DSK FILE FAVAIL RE F LR 13
EXEC RAMISIN SCRR FAVAIL
DETACH RAMIS ALL BRIEF
```

SCRR INST (RAMIS PROCEDURE)

```
DEFINE
FILE MODCOURSE
AVG&CLAS/I3 = (CAP1 * UCAP * .Ø1) + .5
NO&INST/I3 = MAX (1.Ø, (AVG&CLAS / SIR) + .5)
TOT&HRS = NO&INST * CHRS
FLAG/I1 = ASN
END
TABLE
FILE MODCOURSE
STORE TOT&HRS  AND FLAG  BY CDP BY SIR BY NME
IF DEL IS 1
RUN
FILE MODCOURSE
SUM ASN AND HOLD BY CDP BY SIR
END
DEFINE
FILE HOLD
INST&HRS/F5.1 = TOT&HRS / ASN
END
TABLE
FILE HOLD
SUM INST&HRS AND SAVE BY CDP BY NME
IF FLAG IS-MORE-THAN Ø
IF NME IS-NOT '                    '
END
```


TABLE IV-4

```
SCRR FACIL          (RAMIS PROCEDURE)

TABLE
FILE MODFACIL
PRINT REQHRS AND SAVE BY CDP BY BLDG BY RM
END
```

```
SCRR FAVAIL         (RAMIS PROCEDURE)

TABLE
FILE MODFACIL
SUM FST*AVHRS AND SAVE BY BLDG BY RM
END
```

```
SCRR IAVAIL         (RAMIS PROCEDURE)

DEFINE
FILE MODINST
DAYS/I4 = YMD (LDTE, 75Ø7Ø1)
FAC/F4.1 = 365 / DAYS
AVAIL / I4 = MAX ((FAC * YTD), 1)
END
TABLE
FILE MODINST
SUM AVAIL AND SAVE BY NME
IF TT IS A1
END
```

```
SCRR CRSE           (RAMIS PROCEDURE)

DEFINE
FILE MODCOURSE
CLASIZE/I3 = (CAP1 * UCAP * .Ø1) + .5
END
TABLE
FILE MODCOURSE
PRINT CLASIZE AND CONV1 AND SAVE BY CDP
IF DEL IS 1
END
```

TABLE IV-4 (CONTINUED)

BLANK EXEC DESCRIPTION

MPS III linear programming software requires that variable names should be eight characters or less in length and also should not contain embedded blanks. Building and room numbers are stored in the data base as two separate fields - a five character building name and a three character room name. Concatenation of these two names will result in a variable containing embedded blanks if the building name is less than five characters.

The purpose of the executive BLANK is to delete all embedded blank characters in the eight character facility designation contained in the files FILE FACIL and FILE FAVAIL.

Initial statements of BLANK EXEC suppress all terminal output and specify libraries to be searched upon execution of subsequent PLI load commands. The input file FAVAIL for the PLI program BLANK is identified as FILE FAVAIL and the PLI program BLANK is executed. Then the input file FACIL for the PLI program CONCAT is identified as FILE FACIL and CONCAT is executed.

BLANK PLI DESCRIPTION. The input file is FILE FAVAIL. The first eight characters of each thirteen character record are the building number and the room number. BLANK reads each record and determines the index (I) of the first blank occurring in the building number. The first I-1 characters of the record are then substituted for the building number in the eight character building/room designation. The room number from the input record is then concatentated to the I-1 character building number in the output record. The output record which now contains no embedded blanks is then substituted for the input record in FILE FAVAIL.

CONCAT PLI DESCRIPTION. The only difference between CONCAT and BLANK is the name of the input file and the input file record format. CONCAT removes embedded blanks from the eight character building/room designator located in position five of each 22 character record in FILE FACIL.

BLANK EXEC FLOWCHART

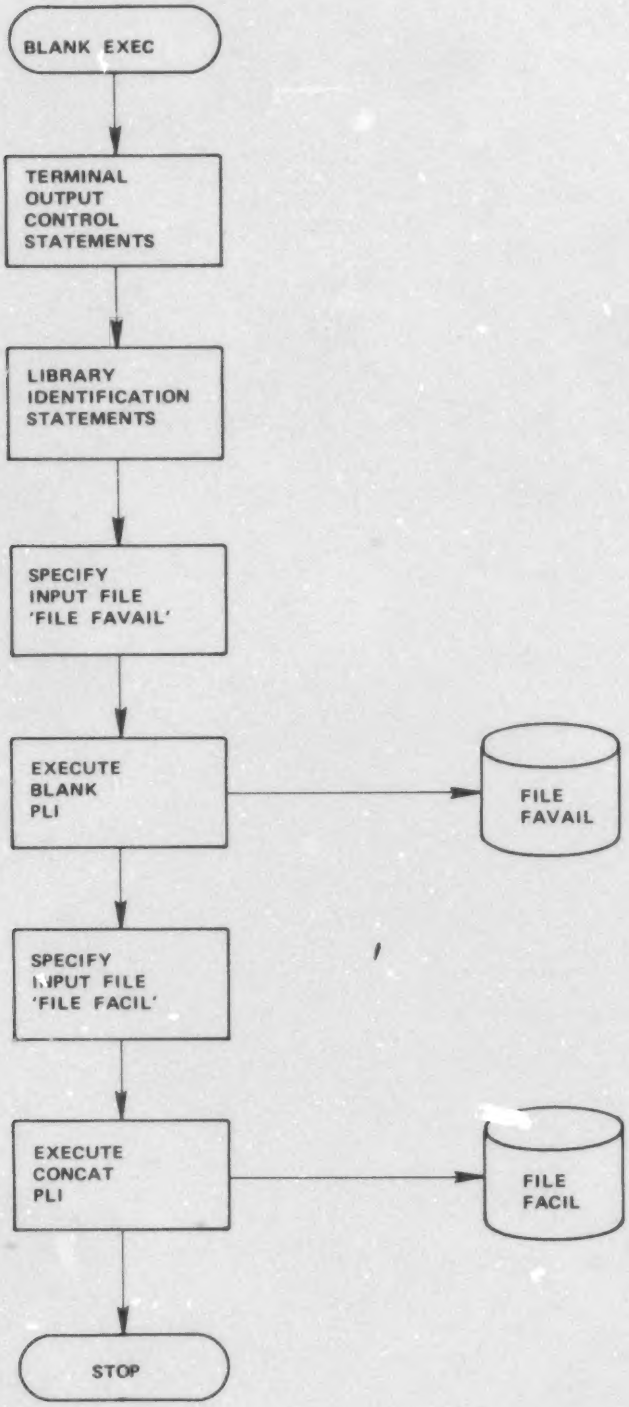


FIGURE IV-4

CHART TITLE - PROC BLANK OPTIONS (MAIN)

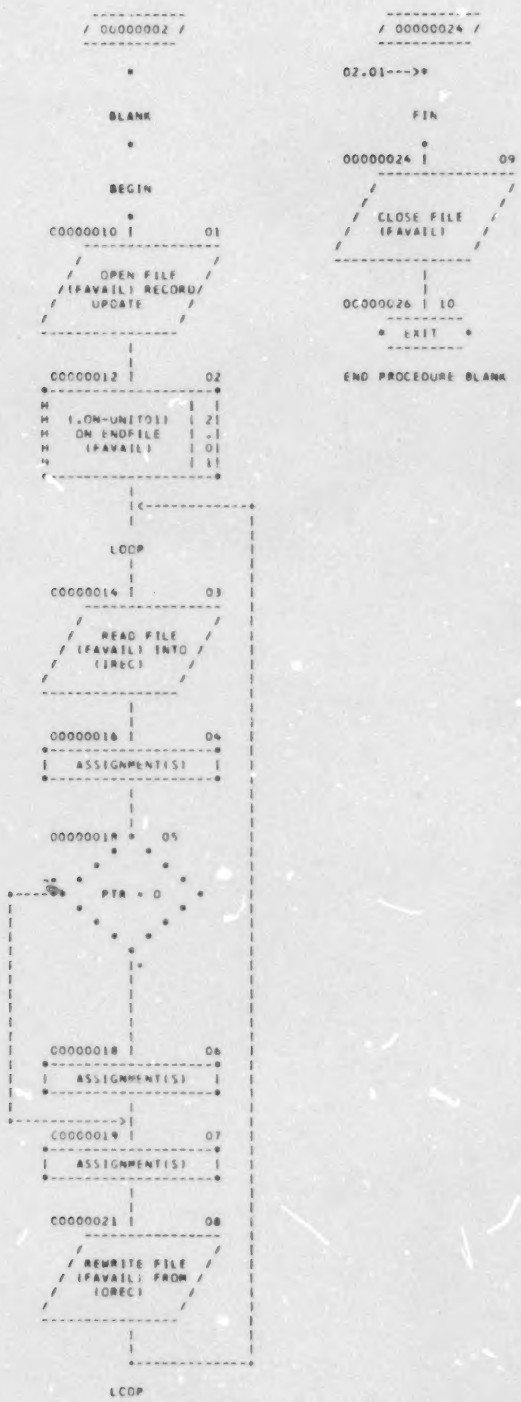


FIGURE IV-4 (CONTINUED)

```
                                    AUTOFLOW CHART SET - FINAL    PL/1 -BLANK- FLOWCHARTS          PAGE  [illegible]

CHART TITLE - ON-UNIT ACTION BLOCKS

                    -----------
                   / 00000012 /
                   -----------

                01.02--->*

               .ON-UNIT01

                         *
                00000012 |  NOTE  01
               * * * * * * * * * * *
               *    ON ENDFILE     *
               *     (FAVAIL)      *
               * * * * * * * * * * *
                         |
                         |
                         |
                        ...
                      .     .
                      . 1.09.
                      .     .
                        ...

                        FIN

                END .ON-UNIT01
```

FIGURE IV-4 (CONTINUED)

BLANK PROGRAM LISTINGS

```
BLANK    EXEC
         VP/CSS

&STACK KT
&TYPE OFF
GLOBAL MACLIB
GLOBAL TXTLIB
GLOBAL LNKLIB
GLOBAL TXTLIB PLILIB SYSLIB
GLOBAL LNKLIB PLILIB SYSLIB
FILEDEF FAVAIL DSK FILE FAVAIL RECFM F LRECL 12
LOAD BLANK (CLEAR LIBE) PLILIB
START
FILEDEF FACIL DSK FILE FACIL RE F LR 22
LOAD CONCAT (CLEAR LIBE) PLILIB
START
```

TABLE IV-5

```
INPUT LISTING                                    AUTOFLOW CHART SET - FINAL    PL/1 -BLANK- FLOWCHARTS

PL/I MODULE        (LIST,NAMSQ,SOURCSEQ)

 CARD NO        ****                          CONTENTS                                    ****

      OPTION PACKIF,COMPRESS                                                          00000100
      BLANK: PROC OPTIONS (MAIN) ;                                                    00000200
              /* READ FILE FAVAIL & ELIM EMBEDDED BLANKS FROM BLDG/RM */               00000300
                                                                                      00000400
              DECLARE (IREC,OREC) CHAR (13),                                          00000500
                      BLDG        CHAR (5)  DEF IREC POS (1),                         00000600
                      RM          CHAR (3)  DEF IREC POS (6),                         00000700
                      BLDGRM      CHAR (8)  DEF OREC POS (1);                         00000800
                                                                                      00000900
      BEGIN:                                                                          00001000
              OPEN FILE (FAVAIL) RECORD UPDATE;                                       00001100
              ON ENDFILE (FAVAIL) GO TO FIN;                                          00001200
                                                                                      00001300
      LOOP:                                                                           00001400
              READ FILE (FAVAIL) INTO (IREC) ;                                        00001500
              OREC = IREC ;                                                           00001600
              PTR = INDEX (BLDG,' ') ;                                                00001700
              IF PTR = 0 THEN PTR = 6 ;                                               00001800
              PTR = PTR - 1 ;                                                         00001900
              BLDGRM = SUBSTR (BLDG,1,PTR) || RM ;                                    00002000
              REWRITE FILE (FAVAIL) FROM (OREC) ;                                     00002100
               GO TO LOOP ;                                                           00002200
                                                                                      00002300
      FIN :                                                                           00002400
              CLOSE FILE (FAVAIL) ;                                                   00002500
              END BLANK ;
```

TABLE IV-5 (CONTINUED)

375

INTERFACE (IF1, IF2, IF3, and IF4) EXECS OVERVIEW

The purpose of the four interface executives IF1, IF1, IF3, and IF4 is to create a card image input data file to be processed by the MPS III linear programming routines. The linear programming input data file consists of data cards that are grouped according to data types or function. Each such group is called a segment. A segment is introduced and identified in the file by an indicator card containing the segment name. Data cards containing the data for the segment follow the indicator card.

The first card in the input data file is a NAME card containing the file name FTCNORVA. The file FTCNORVA contains ROWS, COLUMNS, RHS and BOUNDS segments. The purpose of each segment is defined below:

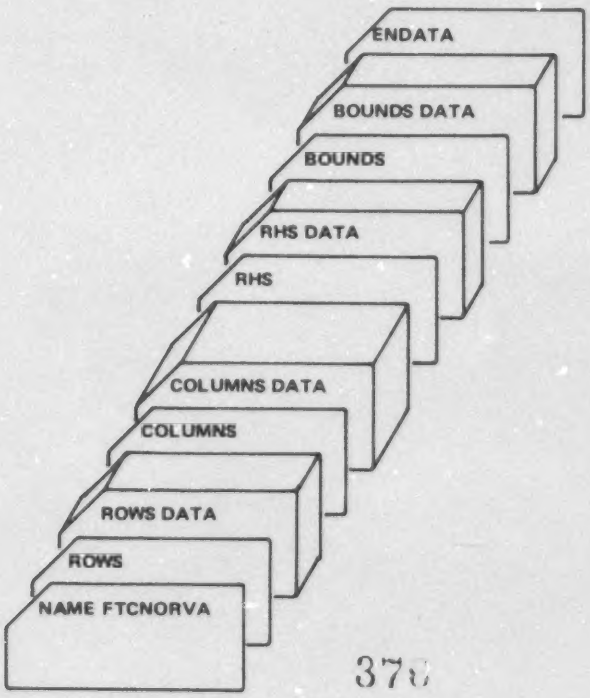
A. ROWS Segment: defines row names and types. The NAME card and the ROWS segment is generated by IF1 EXEC.

B. COLUMNS Segment: defines column names and matrix coefficients. The COLUMNS segment is generated by IF2 EXEC.

C. RHS Segment: defines right-hand-side names and coefficients. The RHS segment is generated by IF3 EXEC.

D. BOUNDS Segment: defines the column bounds set. The BOUNDS segment is generated by IF4 EXEC.

The organization of the LP data input file FTCNORVA is shown below.



DATA BASE INTERFACE EXEC FLOWCHART

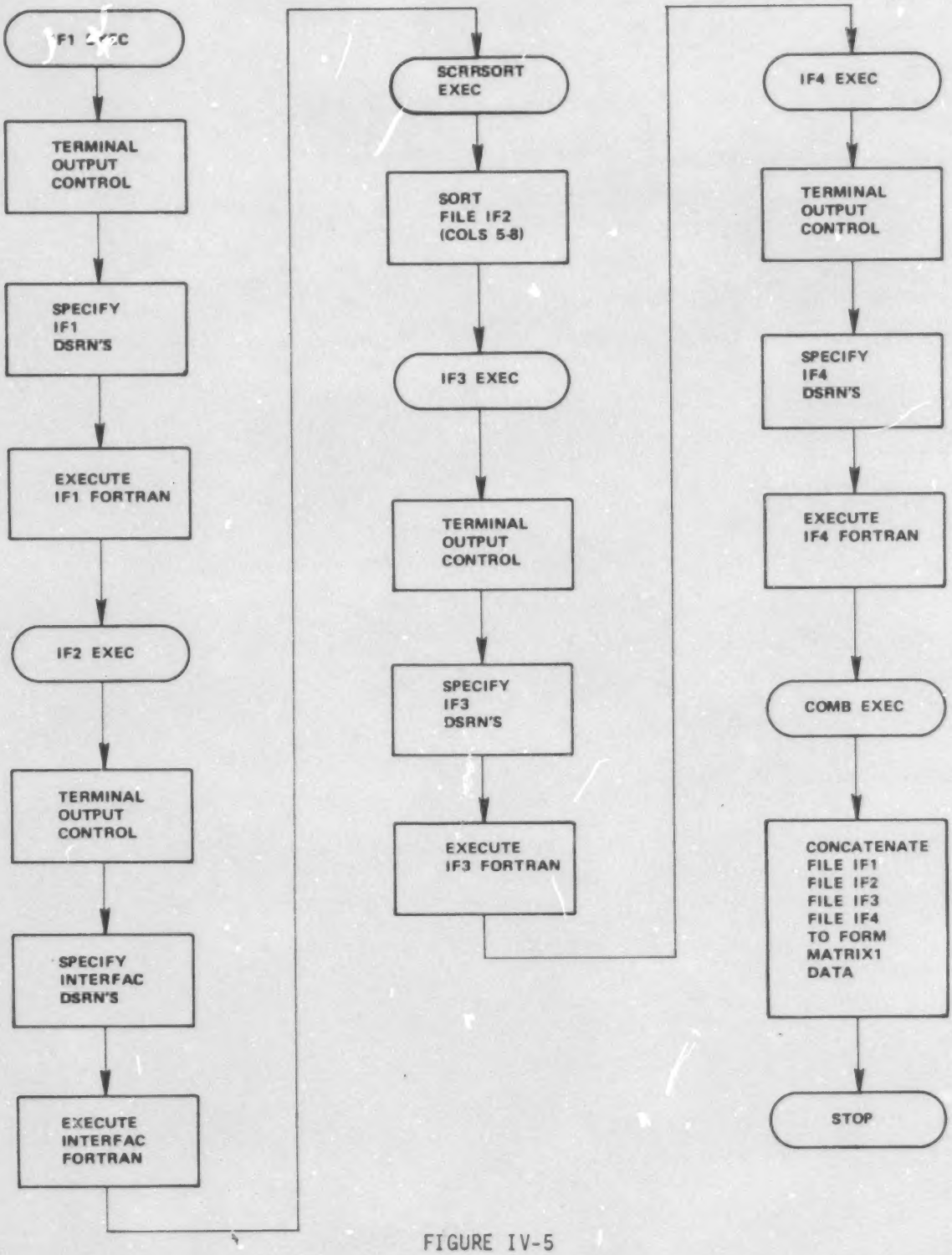


FIGURE IV-5

IF1 EXEC DESCRIPTION

The purpose of IF1 EXEC is to generate both the file name card and the ROW segment of the LP data file. The segment consists of a ROWS indicator card followed by one ROW data card for each row to be defined. The ROW data card contains both the type (N = unconstrained row, for the objective function; or L= less-than-or-equal-to row, for all other rows in the SCRR model LP input matrix) and the row name. One row is defined for each specific resource requirement. The row name will be either an instructor name or a facility name. The ROW data card format is shown below:

CARD COLUMNS	2	5-12
DATA CONTENTS	L or N	ROW NAME

Initial IF1 EXEC statements suppress all terminal output. Then Data Set Reference Numbers (DSRN) are associated with each of the input and output files as follows:

	DSRN	FILENAME	FILETYPE
Input	1	FILE	IAVAIL
	2	FILE	FAVAIL
Output	6	FILE	IF1

Then IF1 FORTRAN IS executed.

IF1 FORTRAN DESCRIPTION. IF1 FORTRAN first writes the LP input file NAME card, the ROWS segment indicator card, and the objective function ROW data card on output file FILE IF1. Then the first eight characters of each instructor name are read from FILE IAVAIL. One ROW data card image is written on FILE IF1 for each instructor name. Each ROW data card image has an L in column 2 and the instructor names in columns 5 - 12. After the last instructor name has been read from FILE IAVAIL and a corresponding data card written on FILE IF1, IF1 FORTRAN reads facility names from FILE FAVAIL and creates a ROW data card for each facility on FILE IF1. IF1 FORTRAN concludes by writing the COLUMNS segment indicator card as the final card image on FILE IF1.

AUTOFLOW CHART SET - FINAL -IF1- FLOWCHARTS

CHART TITLE - PROCEDURES

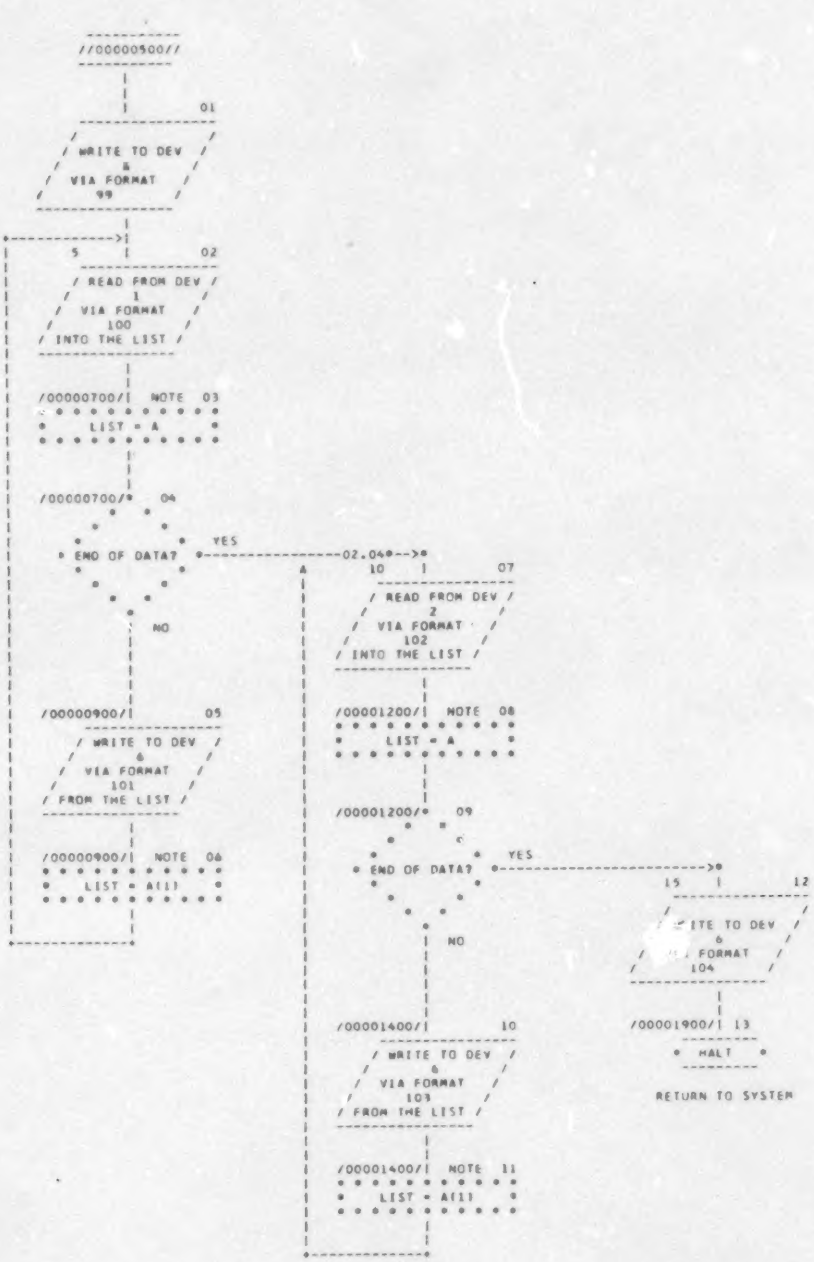


FIGURE IV-6

AUTOFLOW CHART SET - FINAL -IF1- FLOWCHARTS

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      REAL*8 A/2/
99    FORMAT ('NAME',10X,'FTCNORVA'/'ROWS'/' N  THRUPUT')
100   FORMAT (A8,8X,I4)
101   FORMAT (' L  ',A8)
102   FORMAT (A8,I5)
103   FORMAT (' L  ',A8)
104   FORMAT ('COLUMNS')
```

FIGURE IV-6 (CONTINUED)

IF1 PROGRAM LISTINGS

```
IF1       EXEC
          VP/CSS

     E OFF
STACK KT
FILEDEF 1 DSK FILE IAVAIL
FILEDEF 2 DSK FILE FAVAIL
FILEDEF 6 DSK FILE IF1 RE F LR 80
RUN IF1
```

TABLE IV-6

INPUT LISTING AUTOFLOW CHART SET - FINAL -IF1- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
      REAL*8 A(2)                                                       00000100
C                                                                       00000200
C**GENERATE NAME & ROW DATA FOR LP INPUT                                00000300
C                                                                       00000400
      WRITE (6,99)                                                      00000500
   99 FORMAT ('NAME',10X,'FTCNORVA'/'ROWS'/' N  THRUPUT')               00000600
    5 READ (1,100,END = 10) A                                           00000700
  100 FORMAT (A8,8X,I4)                                                 00000800
      WRITE (6,101) A(1)                                                00000900
  101 FORMAT (' L  ',A8)                                                00001000
      GO TO 5                                                           00001100
   10 READ (2,102,END = 15) A                                           00001200
  102 FORMAT (A8,I5)                                                    00001300
      WRITE (6,103) A(1)                                                00001400
  103 FORMAT (' L  ',A8)                                                00001500
      GO TO 10                                                          00001600
   15 WRITE (6,104)                                                     00001700
  104 FORMAT ('COLUMNS')                                                00001800
      STOP                                                              00001900
      END                                                               00002000
```

TABLE IV-6 (CONTINUED)

IF2 EXEC DESCRIPTION.

The purpose of IF2 EXEC is to generate the COLUMN segment of the LP data file. The COLUMN segment consists of a COLUMNS indicator card followed by at least one COLUMN data card image for each course in the RAMIS course file MODCOURSE. Each COLUMN data card image must contain the column name (course CDP number), the row name (instructor or facility name), and the matrix element associated with that column and row. A second row name and matrix element on the same COLUMN card is optional. The COLUMN data card format is shown below:

CARD COLUMNS	5-12	15-22	23-36	40-47	50-61
DATA CONTENTS	COLUMN NAME	ROW NAME	MATRIX ELEMENT	ROW NAME	MATRIX ELEMENT

OPTIONAL

Initial IF2 EXEC statements suppress all terminal output. Then Data Set Reference Numbers (DSRN) are associated with each of the input and output files as follows:

	DSRN	FILENAME	FILETYPE
Input	1	FILE	INST
	2	FILE	FACIL
	3	FILE	CRSE
Output	6	FILE	IF2

Then INTERFAC FORTRAN is executed.

INTERFAC FORTRAN DESCRIPTION. INTERFAC FORTRAN formats and writes the COLUMN segment of the LP data file on file IF2 using the data contained in the three files FILE INST, FILE FACIL, and FILE CRSE. INTERFAC minimizes the size of the COLUMN segment by writing two row names and matrix elements on each COLUMN data card image written for columns which have two or more matrix elements.

After initializing a five column array and several indexing variables, the CDP number, instructor name and contact hours are read from FILE INST. The CDP number is compared with the last CDP number entered into column one of the array.

If the CDP numbers do not match, the CDP number, instructor name and contact hours are written in the first three columns of the next array row. If the CDP numbers match, the contents of column four are examined. If column four contains a zero, then the instructor name and contact hours are written in columns four and five. If column four is not zero, then the CDP number, instructor name and contact hours are written in the first three columns of the next row. After the last record from FILE INST has been read and entered into the array, the array contents are formatted and written on FILE IF2.

The array and indexing variables are then reset and the above procedure is repeated for FILE FACIL. After the last record of FILE FACIL has been read, the array contents are again formatted and written on FILE IF2.

FILE CRSE contains only one record for each CDP number. INTERFAC FORTRAN reads FILE CRSE, then formats and writes the CDP number and class size on FILE IF2.

AUTOFLOW CHART SET - FINAL -INTERFAC- FLOWCHARTS

CHART TITLE - PROCEDURES

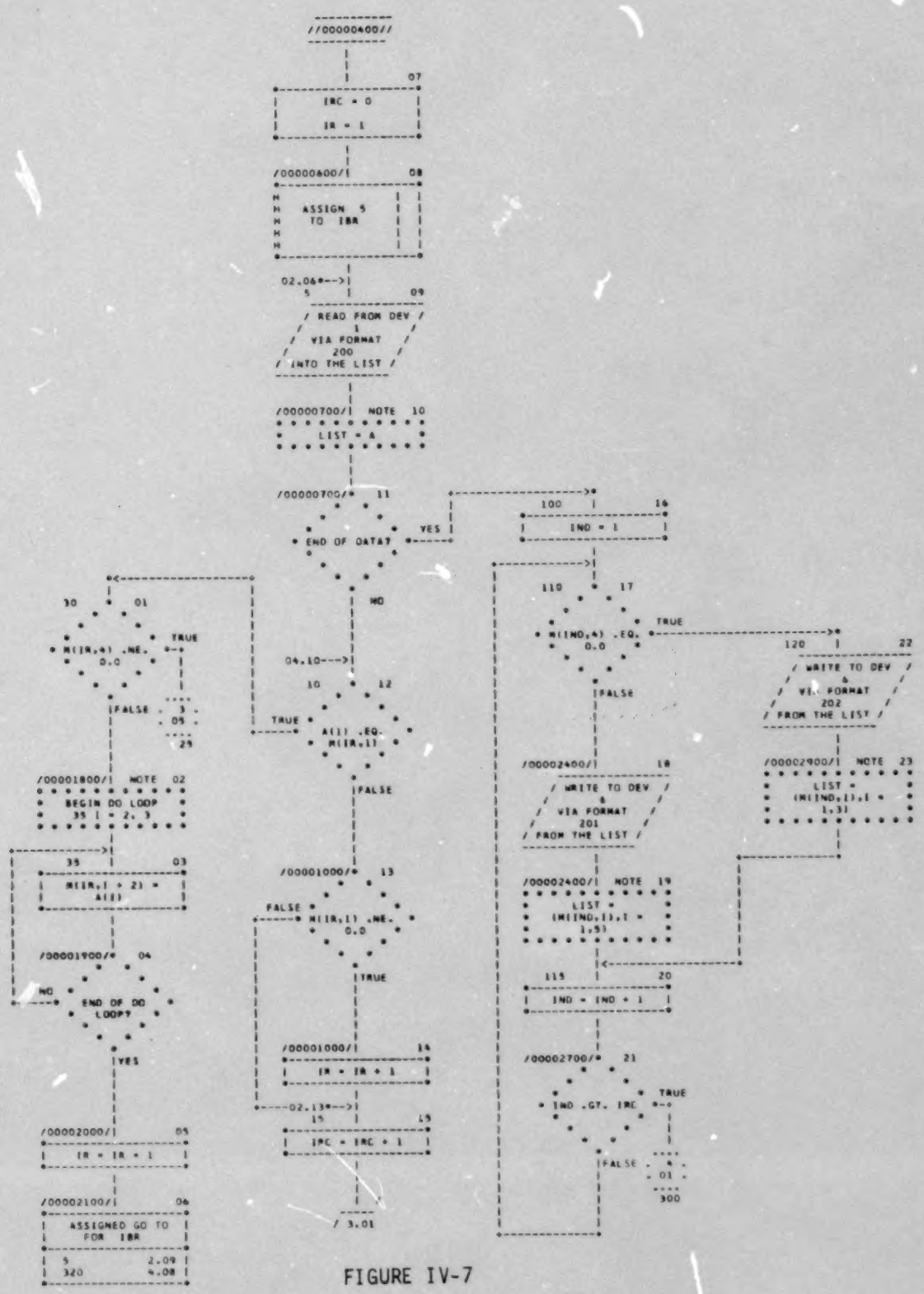


FIGURE IV-7

AUTOFLOW CHART SET - FINAL -INTERFAC- FLOWCHARTS

CHART TITLE - PROCEDURES

```
 02.15--->*
/00001200/|  NOTE  01
* * * * * * * * * * *
*   BEGIN DO LOOP   *
*    20 I = 1, 3    *
* * * * * * * * * * *
          |
*------------>|
|      20    |        02
|   *-------------------*
|   |  N(IR,I) = A.I)   |
|   *-------------------*
|            |
|            |
|            |
|   /00001300/*   03
|          *   *
|        *       *
|   NO  *         *
*-----*   END OF DO   *
        *   LOOP?   *
          *       *
            *   *
              *
              |YES
              |
              |
              |
              |
/00001400/|        04
*-------------------*
|   ASSIGNED GO TO  |
|     FOR  IBR      |
*-------------------*
|  5           2.09 |
|  320         4.08 |
*-------------------*


        ----------
       /   25    /
       ----------
          |
 02.01--->|
          |        05
*-------------------*
|    IR = IR + 1    |
*-------------------*
          |
          |
          |
         ...
        .   .
        . 2.15.
        .   .
         ... 15
```

FIGURE IV-7 (CONTINUED)

AUTOFLOW CHART SET - FINAL -INTERFAC- FLOWCHARTS

CHART TITLE - PROCEDURES

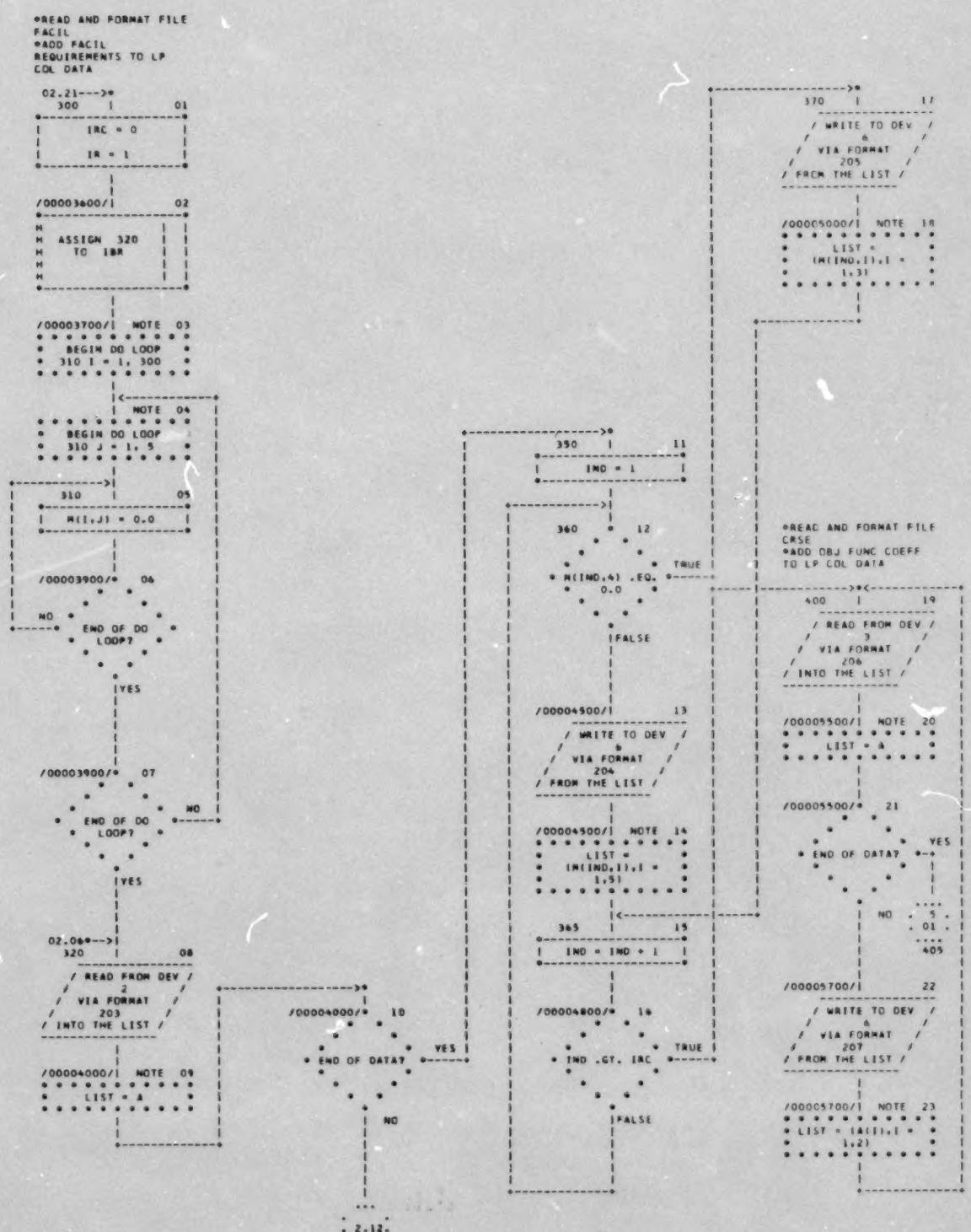


FIGURE IV-7 (CONTINUED)

```
                                        AUTOFLOW CHART SET - FINAL    -INTERFAC- FLOWCHARTS

CHART TITLE - PROCEDURES

                                          ----------
                                         /   405    /
                                         ----------
                                              |
                                    04.21--->|
                                              | 01
                                          ---------
                                         *  HALT   *
                                          ---------

                                        RETURN TO SYSTEM
```

```
                                        AUTOFLOW CHART SET - FINAL    -INTERFAC- FLOWCHARTS

CHART TITLE - NON-PROCEDURAL STATEMENTS


                  REAL*8 A(3),M(300,5)/1500*0.0/
        200       FORMAT (A4,A8,8X,F5.1)
        201       FORMAT (4X,A4,6X,A8,9X,F5.1,3X,A8,9X,F5.1)
        202       FORMAT (4X,A4,6X,A8,9X,F5.1,25X)
        203       FORMAT (A4,A8,F6.1,4X)
        204       FORMAT (4X,A4,6X,A8,8X,F6.1,3X,A8,8X,F6.1)
        205       FORMAT (4X,A4,6X,A8,8X,F6.1,25X)
        206       FORMAT (A4,I3,I3)
        207       FORMAT (4X,A4,6X,'THRUPUT',9X,I3)
```

FIGURE IV-7 (CONTINUED)

388

IF2 PROGRAM LISTINGS

```
IF2       EXEC
         VP/CSS

&TYPE OFF
&STACK KT
FILEDEF 1 DSK FILE INST
FILEDEF 2 DSK FILE FACIL
FILEDEF 3 DSK FILE CRSE
FILEDEF 6 DSK FILE IF2 RECFM F LRECL 80
RUN INTERFAC
```

TABLE IV-7

INPUT LISTING AUTOFLOW CHART SET - FINAL -INTERFAC- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
      REAL*8 A(3),M(300,5)/1500*0.0/
C**READ AND FORMAT FILE INST
C**ADD INST REQUIREMENTS TO LP COL DATA
      IRC = 0
      IR = 1
      ASSIGN 5 TO IBR
    5 READ(1,200,END = 100) A
  200 FORMAT (A4,A8,8X,F5.1)
   10 IF (A(1) .EQ. M(IR,1)) GO TO 30
      IF (M(IR,1) .NE. 0.0) IR = IR + 1
   15 IRC=IRC + 1
      DO 20  I=1,3
   20 M(IR,I) = A(I)
      GO TO IBR, (5,320)
   25 IR = IR + 1
      GO TO 15
   30 IF (M(IR,4) .NE. 0.0) GO TO 25
      DO 35  I=2,3
   35 M(IR,I+2) = A(I)
      IR = IR + 1
      GO TO IBR, (5,320)
  100 IND = 1
  110 IF (M(IND,4) .EQ. 0.0) GO TO 120
      WRITE (6,201) (M(IND,I),I =1,5)
  201 FORMAT (4X,A4,6X,A8,9X,F5.1,3X,A8,9X,F5.1)
  115 IND = IND + 1
      IF (IND .GT. IRC ) GO TO 300
      GO TO 110
  120 WRITE (6,202) (M(IND,I), I =1,3)
  202 FORMAT (4X,A4,6X,A8,9X,F5.1,25X)
      GO TO 115
C**READ AND FORMAT FILE FACIL
C**ADD FACIL REQUIREMENTS TO LP COL DATA
  300 IRC = 0
      IR = 1
      ASSIGN 320 TO IBR
      DO 310 I=1,300
      DO 310 J=1,5
  310 M(I,J) = 0.0
  320 READ (2,203,END = 350) A
  203 FORMAT (A4,A8,F6.1,4X)
      GO TO 10
  350 IND = 1
  360 IF (M(IND,4) .EQ. 0.0) GO TO 370
      WRITE (6,204) (M(IND,I),I=1,5)
  204 FORMAT (4X,A4,6X,A8,8X,F6.1,3X,A8,8X,F6.1)
  365 IND = IND +1
      IF (IND .GT. IRC) GO TO 400
      GO TO 360
  370 WRITE (6,205) (M(IND,I),I=1,3)
  205 FORMAT (4X,A4,6X,A8,8X,F6.1,25X)
      GO TO 365
C**READ AND FORMAT FILE CRSE
C**ADD OPJ FUNC COEFF TO LP COL DATA
  400 READ (3,206,END = 405) A
  206 FORMAT (A4,I3,I3)
      WRITE (6,207) (A(I),I=1,2)
  207 FORMAT (4X,A4,6X,'THRUPUT',9X,I3)
      GO TO 400
  405 STOP
      END
```

TABLE IV-7 (CONTINUED)

SCRRSORT EXEC DESCRIPTION

INTERFAC FORTRAN first writes instructor data by course, then facility data by course, and finally course data by course on FILE IF2. The purpose of SCRRSORT is to sort FILE IF2 by course thereby combining all records associated with each course on FILE IF2.

SCRRSORT EXEC utilizes the NCSS SORT command which moves the records from FILE IF2 to .FILE IF2 arranging them in ascending order according to sort key fields 5 - 8 which contain the course CDP number. The filename .FILE is then changed to FILE using the NCSS ALTER command.

SCRRSORT PROGRAM LISTING

```
SCRRSORT EXEC
        VP/CSS


&TYPE OFF
&STACK 5 8
SORT FILE IF2 (BRIEF)
ERASE FILE IF2
ALTER .FILE IF2 P FILE * *
```

TABLE IV-8

IF3 EXEC DESCRIPTION

The purpose of IF3 EXEC is to generate the RHS segment of the LP data file. The RHS segment consists of a RHS indicator card followed by RHS data card images which contain the RHS vector name and RHS elements. RHS elements are identified by row name (instructor or facility name). The RHS data card format is the same as the COLUMN data card format.

Initial IF3 EXEC statements suppress all terminal output. Then DSRN's are specified for each of the input and output files as follows:

	DSRN	FILENAME	FILETYPE
Input	1	FILE	IAVAIL
	2	FILE	FAVAIL
Output	6	FILE	IF3

Then IF3 FORTRAN is executed.

IF3 FORTRAN DESCRIPTION. IF3 FORTRAN first writes the RHS indicator card on the output file FILE IF3. Then two records each containing an instructor name and available hours are read from FILE IAVAIL and are formatted to form a single RHS data card, which is written on FILE IF3. After the last record is read from FILE IAVAIL, two records each containing a facility name and available hours are read from FILE FAVAIL and are formatted to form a single RHS data card, which is also written on FILE IF3. The process is repeated until all records are read from FILE FAVAIL.

AUTOFLOW CHART SEY - FINAL -IF3- FLOWCHARTS

CHART TITLE - PROCEDURES

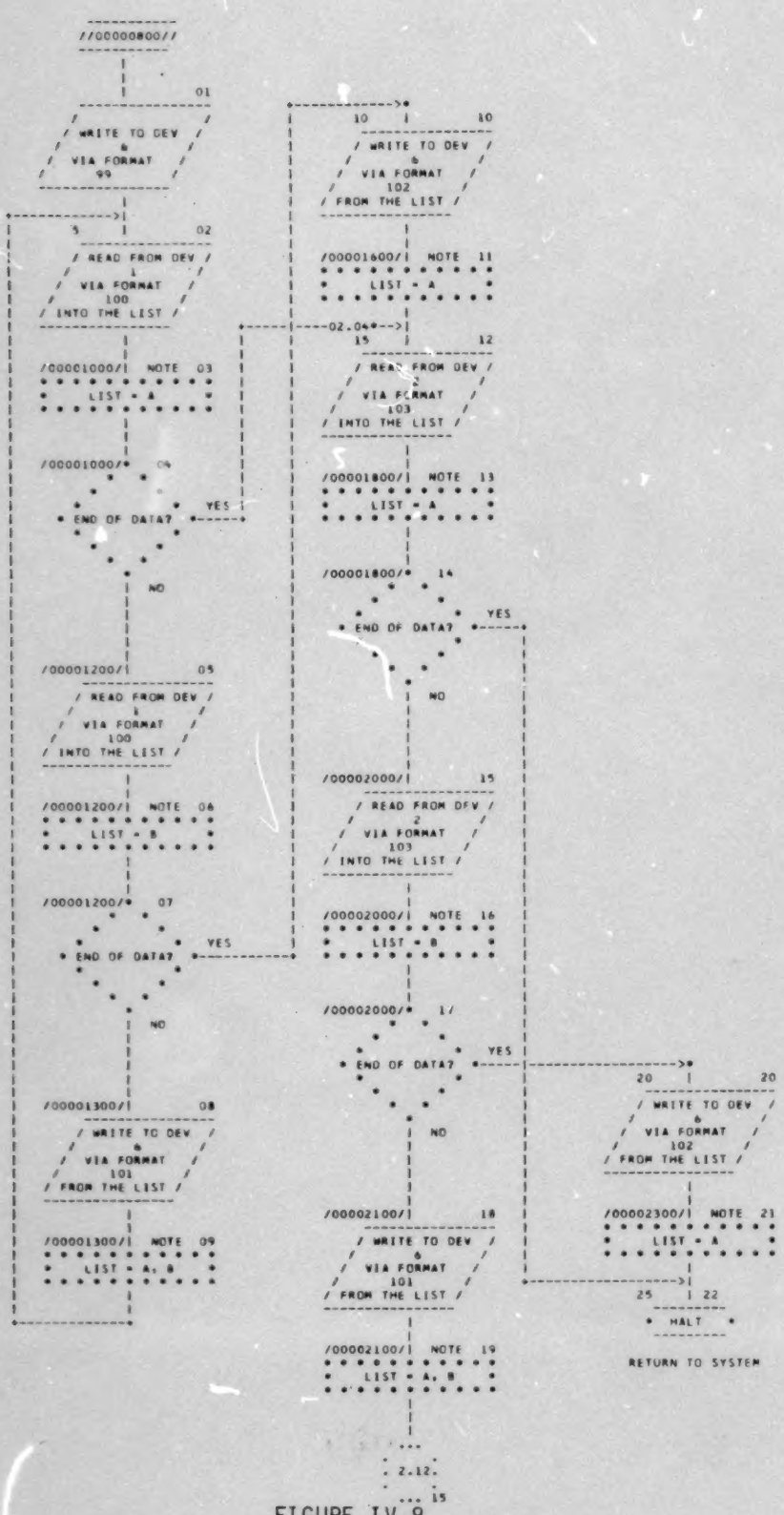


FIGURE IV-8

AUTOFLOW CHART SET - FINAL -IF3- FLOWCHARTS

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      REAL*8 A(2),B(2)
99    FORMAT ('RHS')
100   FORMAT (A8,8X,I4)
101   FORMAT (4X,'AVAIL',5X,A8,8X,I4,5X,A8,8X,I4)
102   FORMAT (4X,'AVAIL',5X,A8,8X,I4)
103   FORMAT (A8,I5)
```

FIGURE IV-8 (CONTINUED)

IF3 PROGRAM LISTINGS

```
IF3       EXEC
         VP/CSS

&TYPE OFF
&STACK KT
FILEDEF 1 DSK FILE IAVAIL
FILEDEF 2 DSK FILE FAVAIL
FILEDEF 6 DSK FILE IF3 RE F LR 80
RUN IF3
```

TABLE IV-9

INPUT LISTING AUTOFLOW CHART SET - FINAL -IF3- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
      REAL*8 A(2),B(2)                                                  00000100
C                                                                       00000200
C**GENERATE RHS DATA SECTION                                            00000300
C  FILE 1 = FILE IAVAIL                                                 00000400
C  FILE 2 = FILE FAVAIL                                                 00000500
C  FILE 6 = FILE IF3                                                    00000600
C                                                                       00000700
      WRITE (6,99)                                                      00000800
   99 FORMAT ('RHS')                                                    00000900
    5 READ (1,100,END = 15) A                                           00001000
  100 FORMAT (A8,8X,I4)                                                 00001100
      READ (1,100,END = 10) B                                           00001200
      WRITE (6,101) A,B                                                 00001300
  101 FORMAT (4X,'AVAIL',5X,A8,8X,I4,5X,A8,8X,I4)                       00001400
      GO TO 5                                                           00001500
   10 WRITE (6,102) A                                                   00001600
  102 FORMAT (4X,'AVAIL',5X,A8,8X,I4)                                   00001700
   15 READ (2,103,END = 25) A                                           00001800
  103 FORMAT (A8,I5)                                                    00001900
      READ (2,103, END = 20) B                                          00002000
      WRITE (6,101) A,B                                                 00002100
      GO TO 15                                                          00002200
   20 WRITE (6,102) A                                                   00002300
   25 STOP                                                              00002400
      END                                                               00002500
```

TABLE IV-9 (CONTINUED)

IF4 EXEC DESCRIPTION

The purpose of IF4 EXEC is to generate the BOUNDS segment of the LP data input. The BOUNDS segment consists of a BOUNDS indicator card followed by BOUNDS data card images that contain lower bounds on the number of annual course convenings for each course. The upper bound is automatically set to $+\infty$. The BOUNDS data card format is shown below:

CARD COLUMN	2-3	5-9	15-22	25-36
DATA CONTENTS	LO	SCHED	COLUMN NAME	BOUND VALUE

Initial IF4 EXEC statements suppress all terminal outout. Then DSRN's are specified for the input and output files as follows:

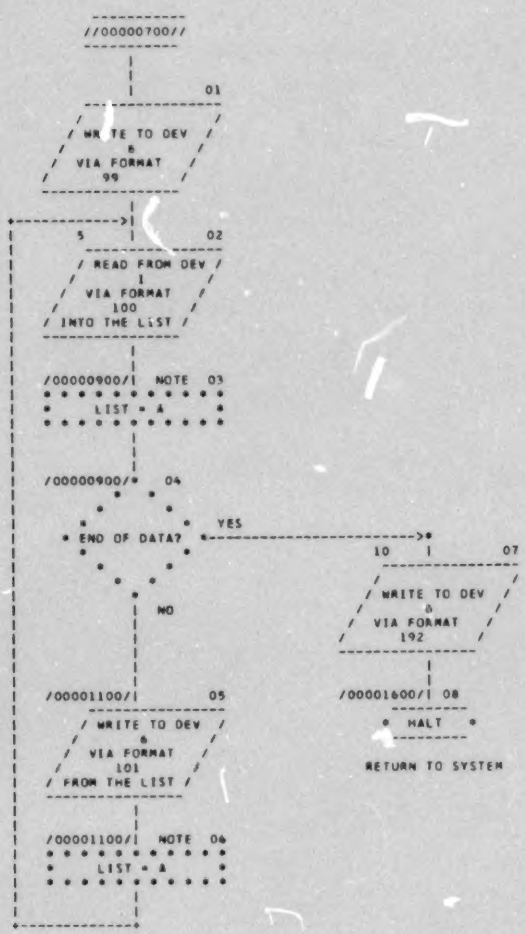
	DSRN	FILENAME	FILETYPE
Input	1	FILE	CRSE
Output	6	FILE	IF4

Then, IF4 FORTRAN is executed.

IF4 FORTRAN DESCRIPTION. IF4 FORTRAN first formats the BOUNDS indicator card on FILE IF4. Then the course CDP numbers and the number of annual course convenings are read from each record of FILE CRSE. The record is reformatted and written on FILE IF4. IF4 FORTRAN concludes by formatting an ENDATA card on FILE IF4.

AUTOFLOW CHART SET - FINAL -IF4- FLOWCHARTS

CHART TITLE - PROCEDURES



AUTOFLOW CHART SET - FINAL -IF4- FLOWCHARTS

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      REAL*8 A(2)
99    FORMAT ('BOUNDS')
100   FORMAT (A4,3X,I3)
101   FORMAT (' LO SCHED',5X,A4,13X,I3)
192   FORMAT ('ENDATA')
```

FIGURE IV-9

IF4 PROGRAM LISTINGS

```
IF4        EXEC
          VP/CSS

&TYPE OFF
&STACK KT
FILEDEF 1 DSK FILE CRSE
FILEDEF 6 DSK FILE IF4 RE F LR 80
RUN IF4
```

TABLE IV-10

INPUT LISTING AUTOFLOW CHART SET - FINAL -IF4- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
      REAL*8 A(2)                                                       00000100
C                                                                       00000200
C**GENERATE BOUNDS DATA SECTION                                         00000300
C  FILE 1 = FILE CRSE                                                   00000400
C  FILE 6= FILE IF4                                                     00000500
C                                                                       00000600
      WRITE (6,99)                                                      00000700
   99 FORMAT ('BOUNDS')                                                 00000800
    5 READ (1,100, END = 10) A                                          00000900
  100 FORMAT (A4,3X,I3)                                                 00001000
      WRITE (6,101) A                                                   00001100
  101 FORMAT (' LO SCHED',5X,A4,13X,I3)                                 00001200
      GO TO 5                                                           00001300
   10 WRITE (6,192)                                                     00001400
  192 FORMAT ('ENDATA')                                                 00001500
      STOP                                                              00001600
      END                                                               00001700
```

TABLE IV-10 (CONTINUED)

COMB EXEC DESCRIPTION

The purpose of COMB EXEC is to combine the four output files created by the interface executives into a single input file suitable for processing by MPS III/OL.

COMB EXEC first uses the NCSS command ERASE to delete the existing copy of the file MATRIX1 DATA from writable disk. Then MATRIX1 DATA is recreated using the NCSS command COMBINE which joins the four files FILE IF1, FILE IF2, FILE IF3 and FILE IF4 to form MATRIX1 DATA.

COMB PROGRAM LISTING

```
COMB      EXEC
          VP/CSS

&TYPE OFF
ERASE MATRIX1 DATA
COMBINE MATRIX1 DATA P FILE IF1 P FILE IF2 P FILE IF3 P FILE IF4 P
```

TABLE IV-11

SCRR1 EXEC DESCRIPTION

The purpose of SCRR1 EXEC is to attach and initialize the MPS III/OL system and execute a series of Terminal Control Language (TCL) statements contained in the SCRR RUN file. MPS III is an advanced Mathematical Programming System (MPS) which has been designed to solve linear programming (LP) problems. The MPS III On-line System (MPS III/OL) is adpated for interactive use under VP/CSS, the National CSS, Inc. time-sharing system. TCL is similar to many computer programming languages (such as FORTRAN), but is oriented specifically to the solution of mathematical programming problems.

Upon entry, existing core size is checked. If core size is less than 320K, a message directing the use to SET CORE 320 is printed and the EXEC is terminated. If minimum core size requirements are met, SCRR1 EXEC then suppresses both offline print and terminal output commands contained within the MPS III/OL system. The file MPSOUT LISTING which contains a detailed record of the LP computational process is then erased. (MPSOUT LISTING is regenerated each time the SCRR model is executed). Output files for both the MPS III LP solution output and the SCRR model output are established by the FILEDEF commands.

The command ATTACH MPSIII accesses a VP/CSS system EXEC file that attaches the necessary disks, loads the Terminal Executor and requests information regarding the location of the scratch desk for work files, the sysin file filename, probfile names, card image filenames, and actfile names. Responses to these requests are stored in a special area of user storage called the CSS stack. The stack containing the MPS III file initialization information is generated by SCRR1 EXEC. The information is read from the stack each time the system EXEC requests file data. The file information is formed into an argument list and a START MPSIII TMLX command is issued. The system prompting messages,

```
MPS III, ON-LINE VERSION, READY
MPS COMMAND:
```

are then printed out at the terminal. At this point the RUN SCRR command, which has also been stacked, is executed. SCRR1 EXEC concludes by printing a message which explains how to obtain a listing of the SCRR model output.

SCRR RUN DESCRIPTION. The purpose of SCRR RUN is to calculate the LP optimal solution and the post-optimal sensitivity analysis for the problem formulated and written on the file MATRIX1 DATA. SCRR RUN is a series of TCL statements which control the execution sequence of the LP problem.

<u>TITLE ' ' AND TERM OFF</u>. These two TCL commands are used to suppress MPS III/OL printed output at the user's terminal.

<u>SET XDONFS = 16</u>. This command is used to alter the value of the MPS III/OL Communications Region (CR) cell XDONFS. XDONFS originally contained the address of the standard procedure which terminates MPS III/OL prior to filing the solution matrix when a no-feasible-solution condition is encountered during optimization. Altering the address of the XDONFS procedure to RUN program statement 16 enables the solution output to be filed for both optimal and non-feasible-solution conditions.

CALL ASSIGN FT04F001 FT04F001 COMM. This statement specifies that MPS III/OL output is to be written on the FORTRAN file 4 in communications format.

CALL PREPOUT FT04F001. This statement positions the communications file FT04F001 to the beginning of the area to receive data and opens that file for output.

CALL ASSIGN FT07F001 FT07F001 CARD. This statement specifies that FORTRAN file 7 is to be written in card-image format.

ROUTE SYSPRINT MPSOUT. This command causes the detailed record of the LP computational process to be written on the file MPSOUT LISTING. All MPS III/OL error messages (when an error condition is encountered) will be stored on MPSOUT LISTING and will not be printed at the terminal during model execution.

SET XDATA = FTCNORVA, XPBNAME = SCRR1. This instruction alters the value of the CR location referenced by the name XDATA to the input data set name FTCNORVA, and alters the value of the location XPBNAME to the problem file name SCRR1. This allows the files to be referenced later by control program statements. The location of both the input data set and the problem file is required by the CONVERT and SETUP procedures.

CALL CONVERT. CONVERT provides the basic means of problem input. The procedure reads the input data, converts them to packed binary format, and writes them on the PROBFILE. The PROBFILE is a disk storage unit that contains the internal representation of the LP problem. While the problem is being generated, comprehensive error checks are made, and the statistics of the model are calculated for use by the SETUP procedure.

SET XOBJ - THRUPUT, XRHS = AVAIL. This instruction alters the value of the XOBJ CR location to the name of the LP objective function THRUPUT, and alters the value of the location XRHS to the name of the right-hand-side vector AVAIL.

CALL SETUP, BOUND = SCHED, MAX. SETUP reads the LP problem created by CONVERT on PROBFILE. The SETUP procedure then analyzes the model's statistics, types of input/output devices, and solution strategy to be used. The available memory is dynamically allocated for efficient problem solution and all work files used by the computational procedures are produced. SETUP also determines the initial LP basic solution. The SETUP parameter, BOUND, indicates the name of the BOUND vector in the original input data. The MAX parameter calls for the maximization of the objective function. The procedure SETUP writes the following output on the file MPSOUT LISTING:

a. Table of files that have been defined
b. Storage allocation tables
c. Table of statistics regarding the matrix variables.

CALL VARIFORM. VARIFORM is the main optimization procedure. It attempts to find a feasible solution, followed by an optimal solution. If the problem has no feasible solution, i.e., when requirements are larger than availability, an appropriate message is printed on MPSOUT LISTING and VARIFORM is terminated. Upon termination of VARIFORM, the control program branches to statement 16 which causes only the SOLUTION output to be written on FORTRAN file 4.

VARIFORM uses a composite algorithm and the revised simplex method. It progresses from the initial basis to the optimal basis by a series of vector interchanges; one vector is introduced into the basis and one is removed.

VARIFORM outputs on MPSOUT LISTING an iteration log which notes for each iteration number, the vector number removed from the basis, the vector number entered into the basis, the reduced cost of the incoming vector, and the sum of remaining infeasiblities or value of the objective function.

CALL SOLUTION FILE FT04F001 RSECTION, 2/4/7, CSECTION, 2/4/5/6. SOLUTION outputs in communications format a summary of the present LP problem and its current solution during the computational procedure if the solution is infeasible, or after the computational procedure if the solution is optimal. SOLUTION writes three arrays named SOLUTION, RSECTION and CSECTION on FORTRAN file 4. The SOLUTION array contains problem identification data. RSECTION array contains four data columns: row name (2), activity (4), slack activity (5), and upper limit (7). CSECTION array also contains four data columns: column name (2), activity (4), input cost (5), and lower limit (6).

CALL RANGE FILE FT04F001 RSECTION, 1/2/3/5, CSECTION, 2/3/4/5. Range is used post-optimally to generate and output an analysis of the current solution. The analysis includes:

a. The effects of changes in class size on optimum convening levels.

b. The impact on student throughput of changing the number of course convenings from optimum level, and the total range of course convenings for which the impact is valid.

c. The impact on student throughput of modfying a resource level and the interval for which this input is valid.

These data are tabulated in five arrays named RANGE, SECTION1, SECTION2, SECTION3, and SECTION4 and written on FORTRAN file 4. RANGE is a single row section containing information identifying the problem, model, and progress of the compution. SECTION1 and SECTION3 are row arrays each containing seven columns of data: row name (1), upper and lower activity range (2), unit cost associated with both upper and lower range value (3), and the limiting process associated with each range value (5). SECTION2 and SECTION4 are column arrays which include nine columns of data: seven columns are identical to those contained in the row arrays; column arrays also contain upper and lower cost values (4).

SKIP 1. This statement causes the next statement in the run file to be bypassed.

CALL PREPIN FT04F001. This statement positions the communications file FT04F001 to the start of the data and opens the file for input.

CSS ALTER SCRRFORM.....CALL USERPROC.....CSS ALTER USERPROC.....This series of statements changes the name of SCRRFORM LNKLIB to USERPROC LNKLIB, executes USERPROC, then changes the LNKLIB file name back to SCRRFORM.

SCRRFORM LNKLIB contains both the relocatable object program for SCRRFORM FORTRAN and the READCOMM library. READCOMM (Read Communications Format) transfers to the SCRRFORM FORTRAN procedures the data previously filed in MPS III Communications format.

EXIT. The EXIT command terminates the MPS III/OL run and returns the user to the VP/CSS command level. The system responds to the EXIT command by printing out "MPSIII TERMINATING".

SCRRFORM FORTRAN DESCRIPTION. The purpose of SCRRFORM FORTRAN is to read the MPS III/OL output which has been filed on FORTRAN file 4 in communications format, reformat the LP output, and write the results on file SCRR OUTPUT.

All MPS III/OL filed outputs are written in communications format, which cannot be read directly by the FORTRAN READ statement. Read Communications Format (READCOMM) subroutines are required to retrieve and transfer the data contained in arrays filed in Communications Format by MPSIII. SCRRFORM utilizes three READCOMM subroutines:

POSITN - positions a communications format file to a specified array.

ARRAY - initializes READCOMM to read the data contained in the array.

VECTOR - transfers a row of elements from the filed array to the calling program.

These subroutines act as an interface between the FORTRAN procedures under the control of the MPSIII executor and the Communications Format subroutines of MPSIII.

Upon entry, SCRRFORM initializes several variables, vectors and arrays. Then the RSECTION array is read from FORTRAN file 4 into the first four columns of the array R. After the last row vector has been read from RSECTION, SCRRFORM searches file 4 for the row-related arrays created by the MPSIII sensitivity analysis, SECTION1 and SECTION3. SECTION1 contains data for rows at their upper limit; SECTION3 contains data for rows at an intermediate level. If the current problem has an infeasible solution, the arrays will not exist, and the program branches to the resource (row) output format routine. If the sensitivity analysis has been performed, SCRRFORM first reads the data from SECTION1 array one row at a time. Row names are stored in column one of the array R and in column one of SECTION1. The row names in both arrays are in sorted sequence. Each row name from SECTION1 is compared with the row name in R until a match is found. Then columns two through seven of SECTION1 are read into columns five through ten of array R. The process is repeated until all rows from SECTION1 have been read into R. SCRRFORM then positions file 4 to the SECTION3 array. Row names are matched and SECTION3 data is transferred to the array R. After the last row has been read from SECTION3 and entered into R, each row of R will have ten columns of data associated with it. SCRRFORM then branches to the resource (row) output format routine, which writes column headings, and reformats the contents of R on output file SCRR OUTPUT.

After the last row of resource data has been formatted and written on the output file, SCRRFORM branches to the column section routines. Data from CSECTION are transferred to a twelve column array C, and data from SECTION2 and SECTION4 arrays (if the sensitivity analysis has been performed) are added to C, using an algorithm similar to that used to construct the row output data. Column headings are generated and the course (column) data output is appended to the resource data output on the file SCRR OUTPUT.

SCRR1 EXEC FLOWCHART

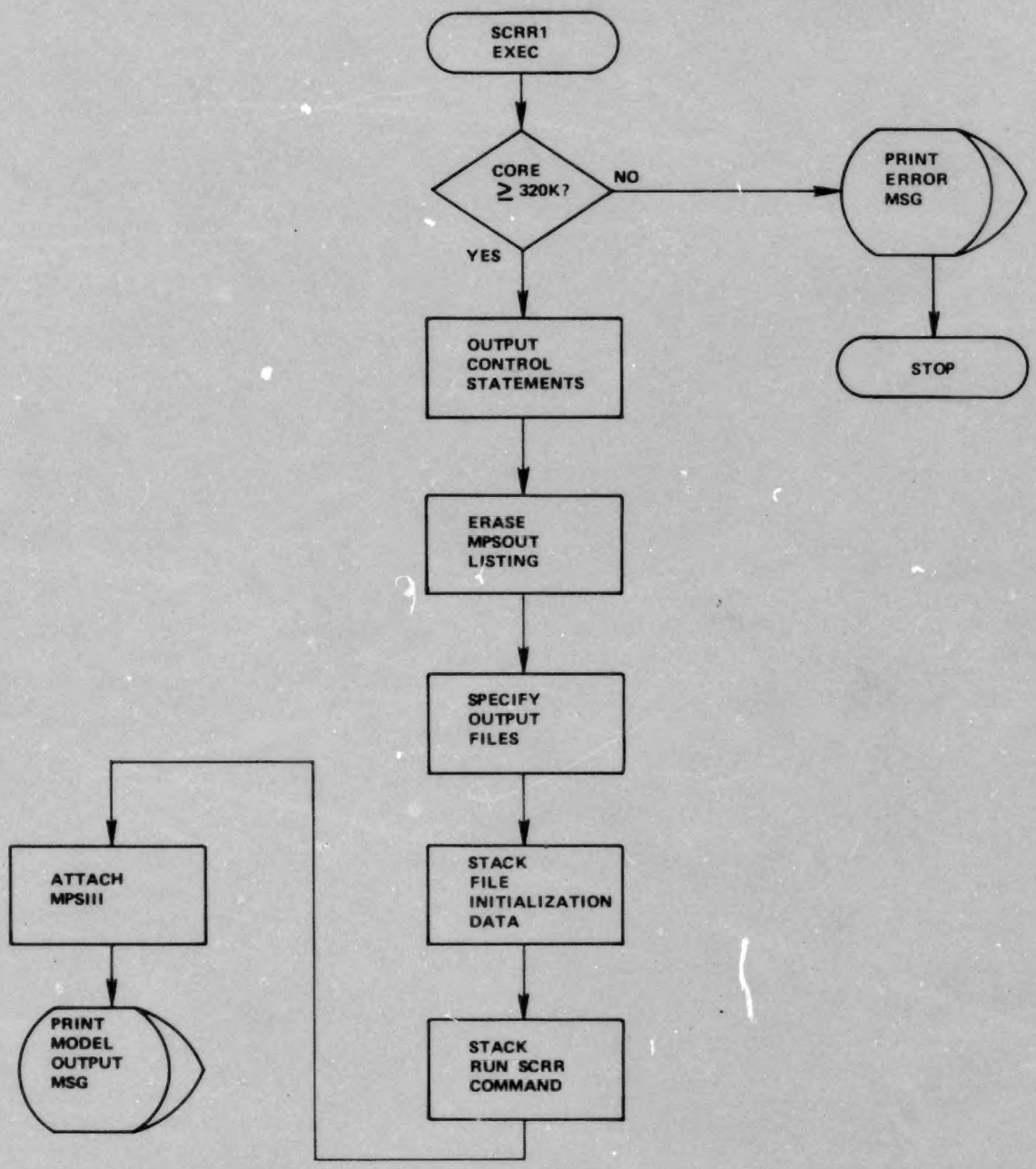


FIGURE IV-10

CHART TITLE - SUBROUTINE FORMAT

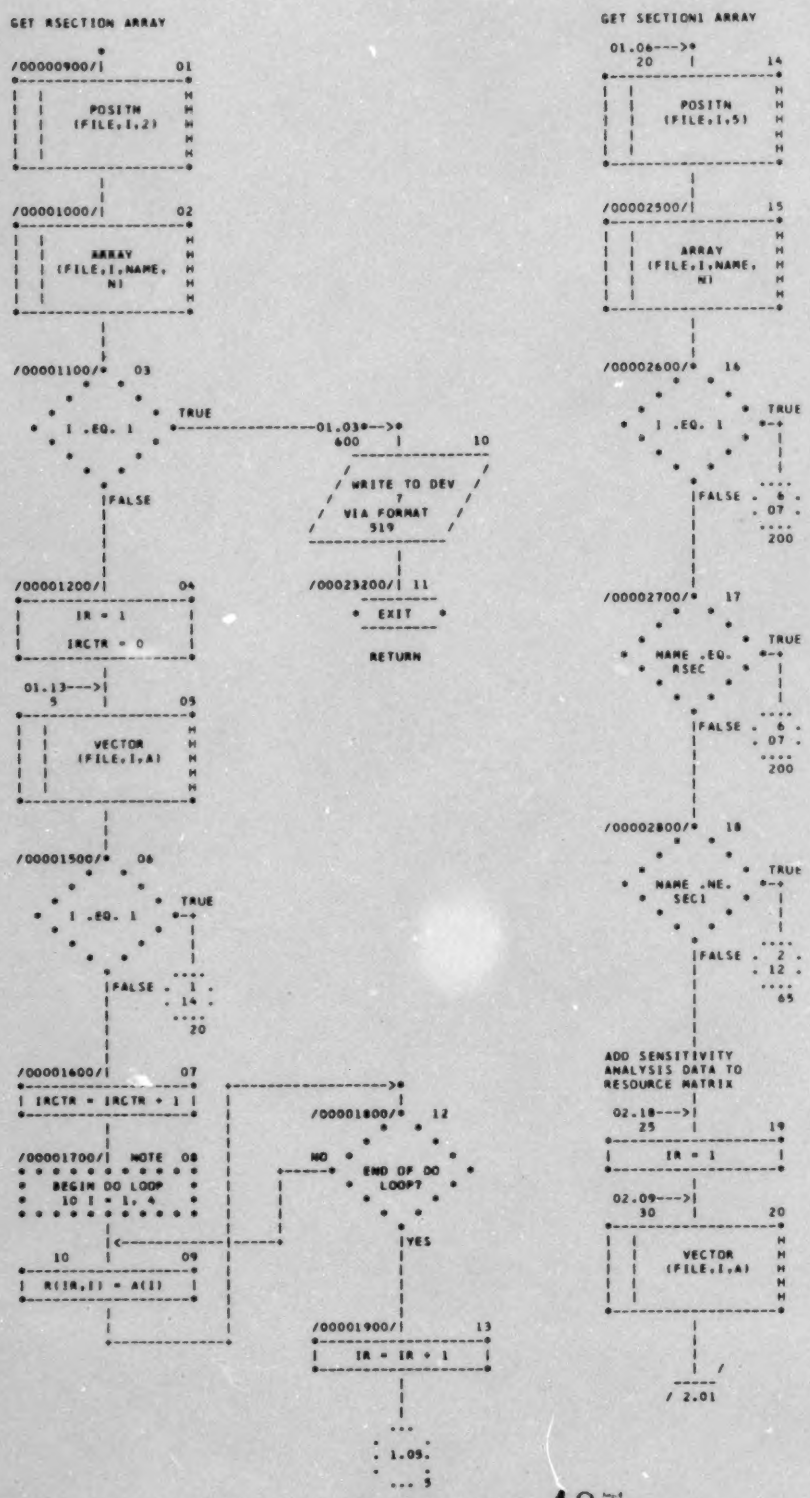
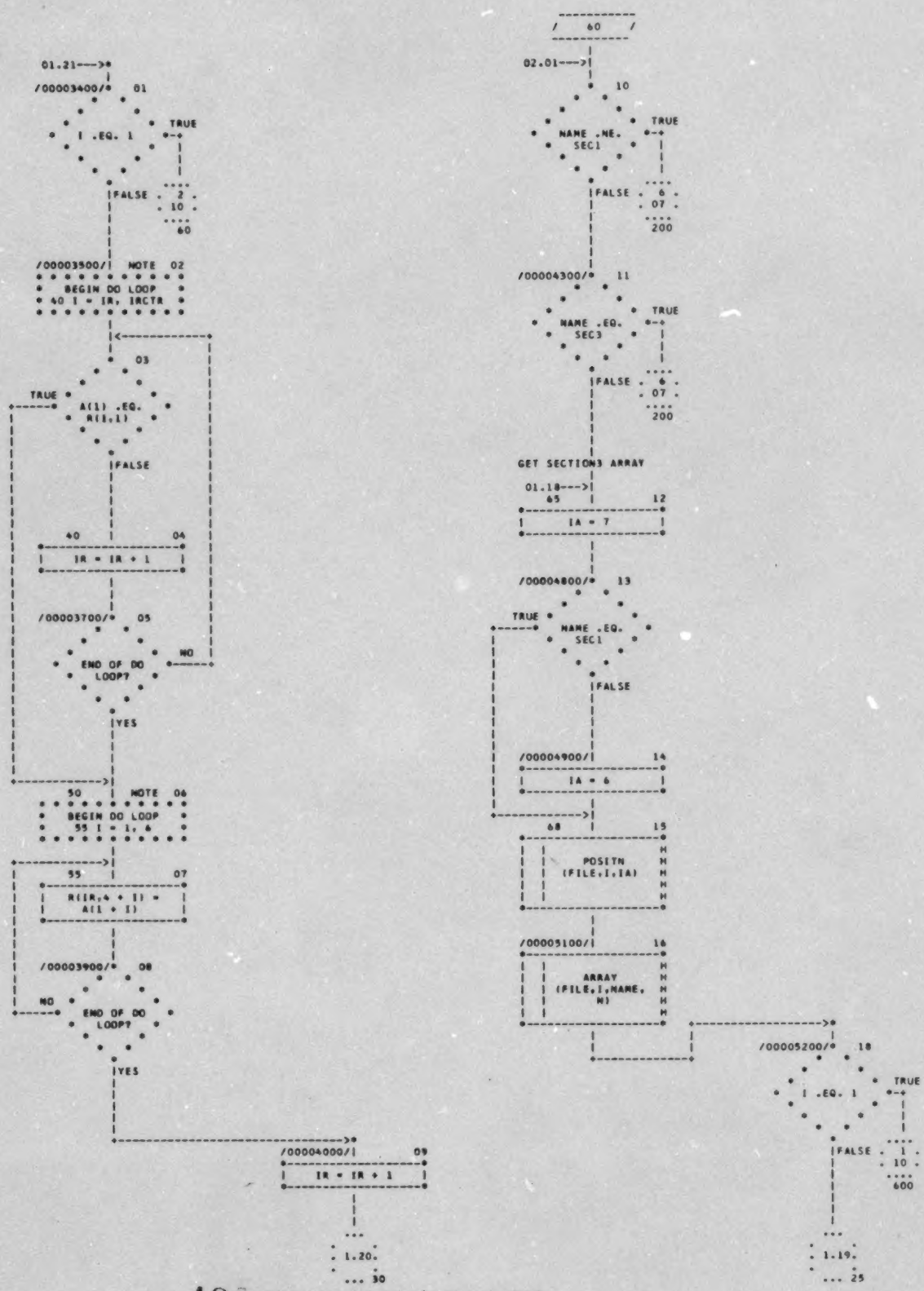


FIGURE IV-10 (CONTINUED)

CHART TITLE - SUBROUTINE FORMAT



406 FIGURE IV-10 (CONTINUED)

CHART TITLE - SUBROUTINE FORMAT

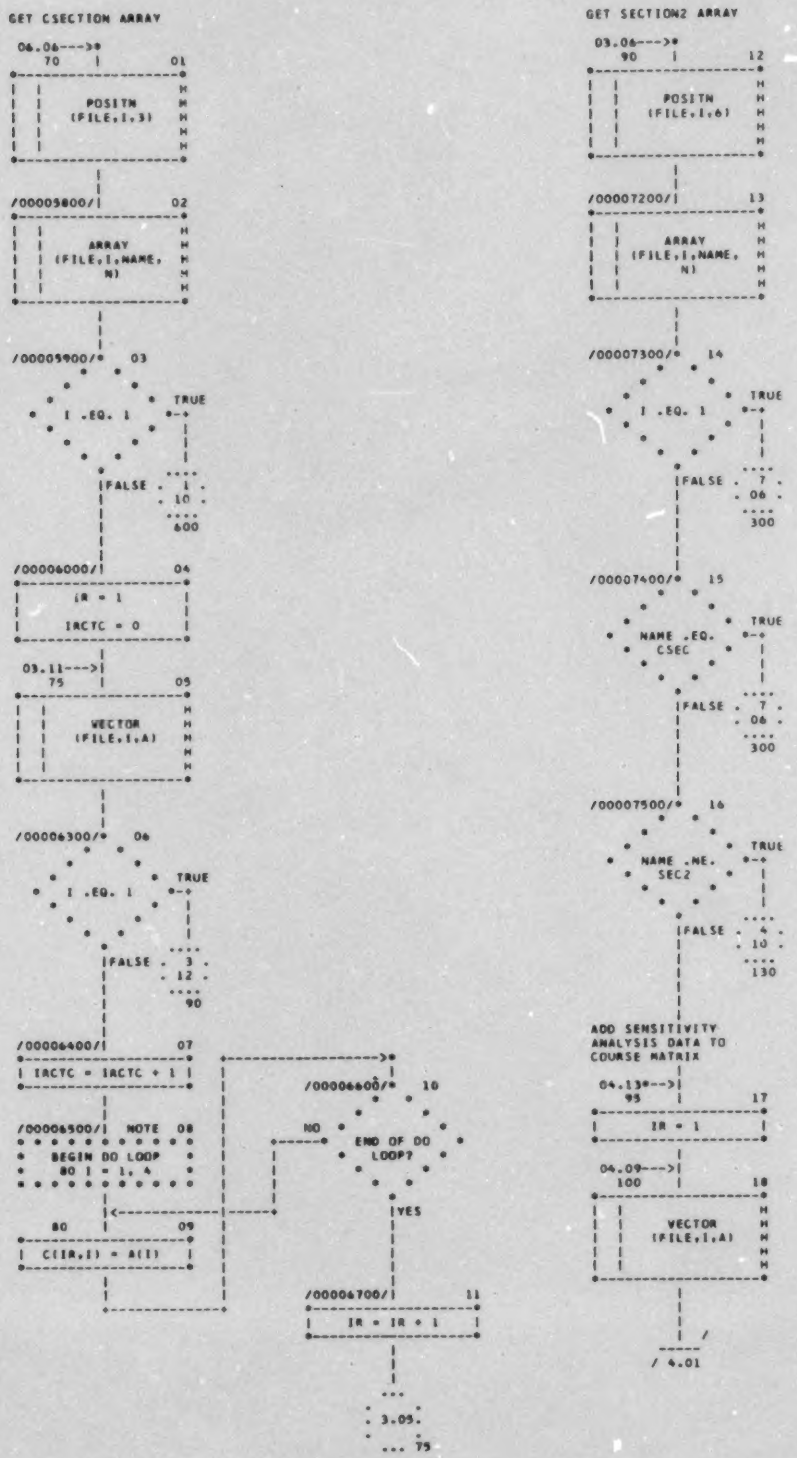


FIGURE IV-10 (CONTINUED)

CHART TITLE - SUBROUTINE FORMAT

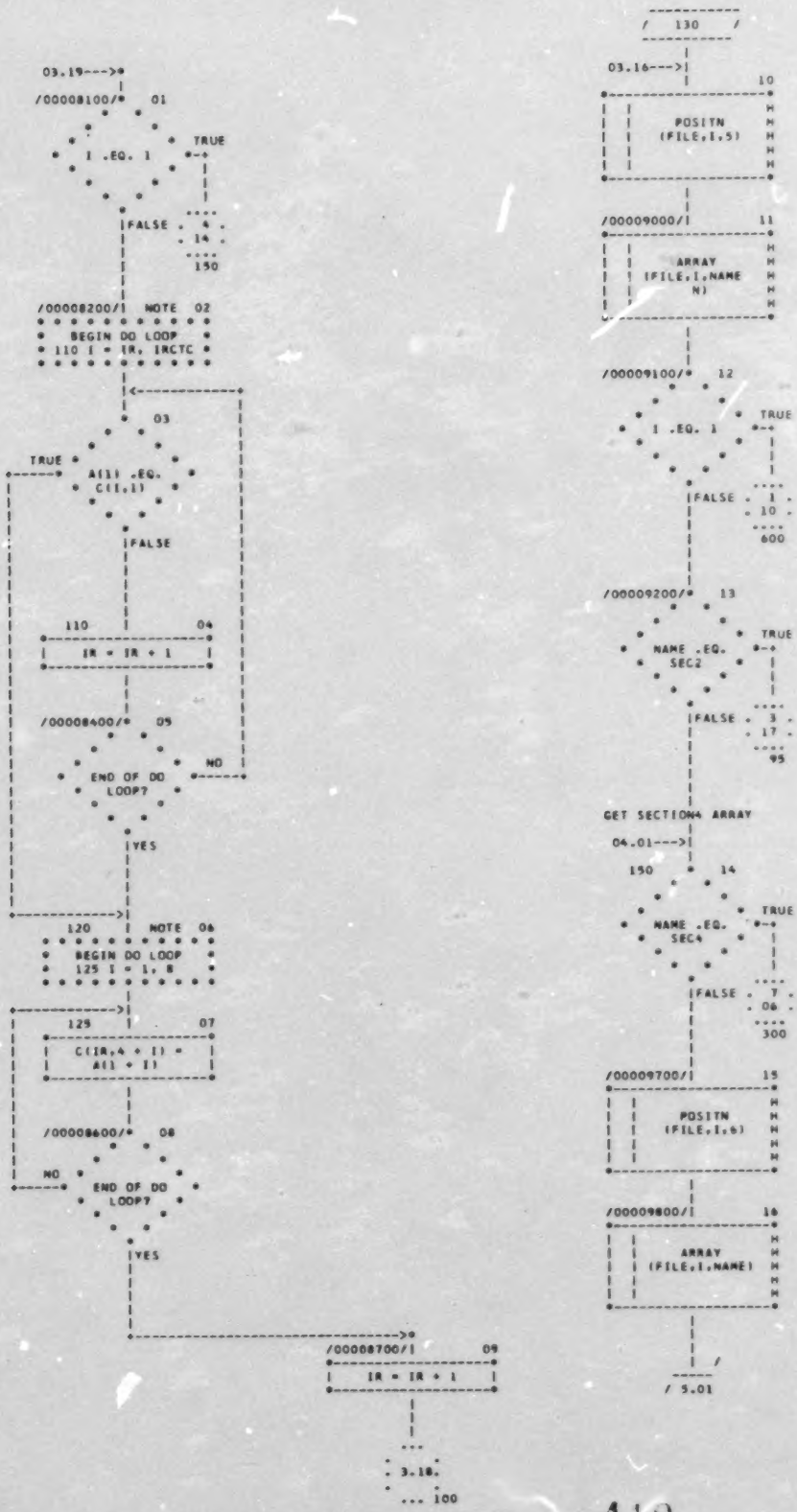


FIGURE IV-10 (CONTINUED)

CHART TITLE - SUBROUTINE FORMAT

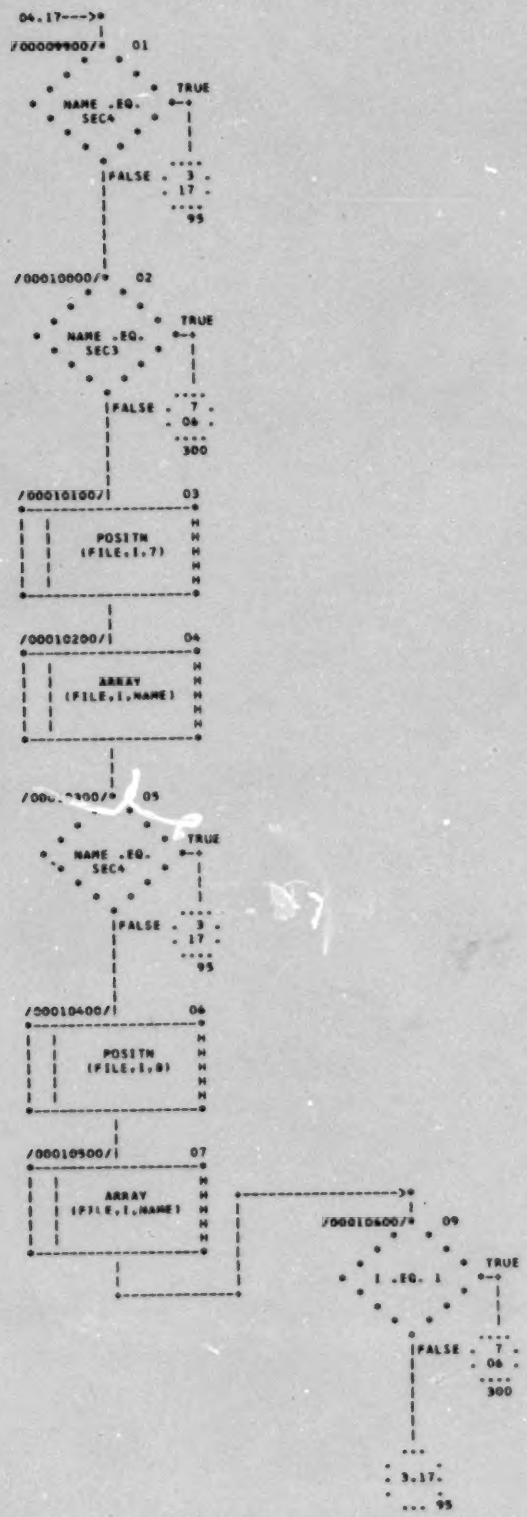


FIGURE IV-10 (CONTINUED)

411

CHART TITLE - SUBROUTINE FORMAT

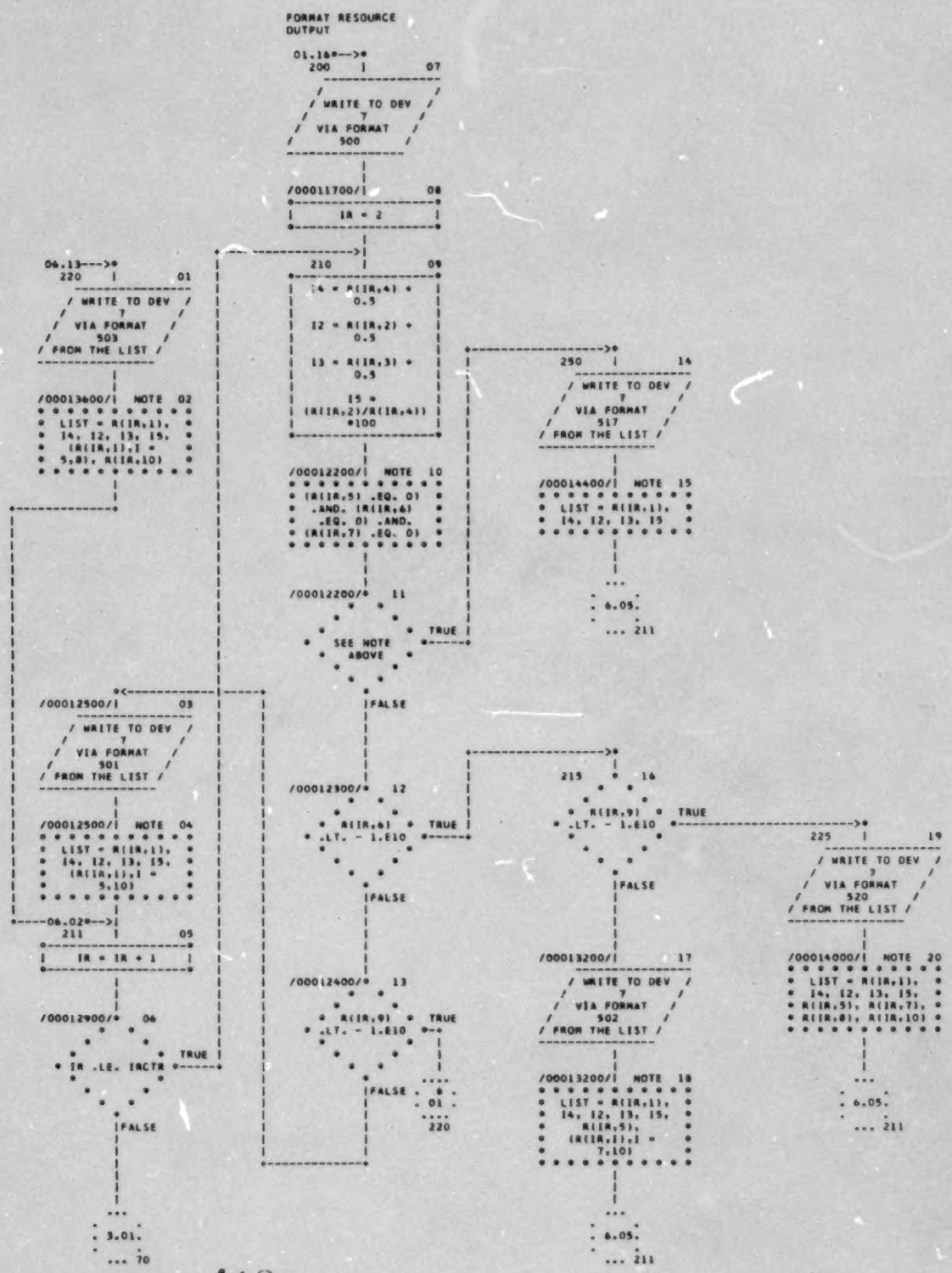


FIGURE IV-10 (CONTINUED)

CHART TITLE - SUBROUTINE FORMAT

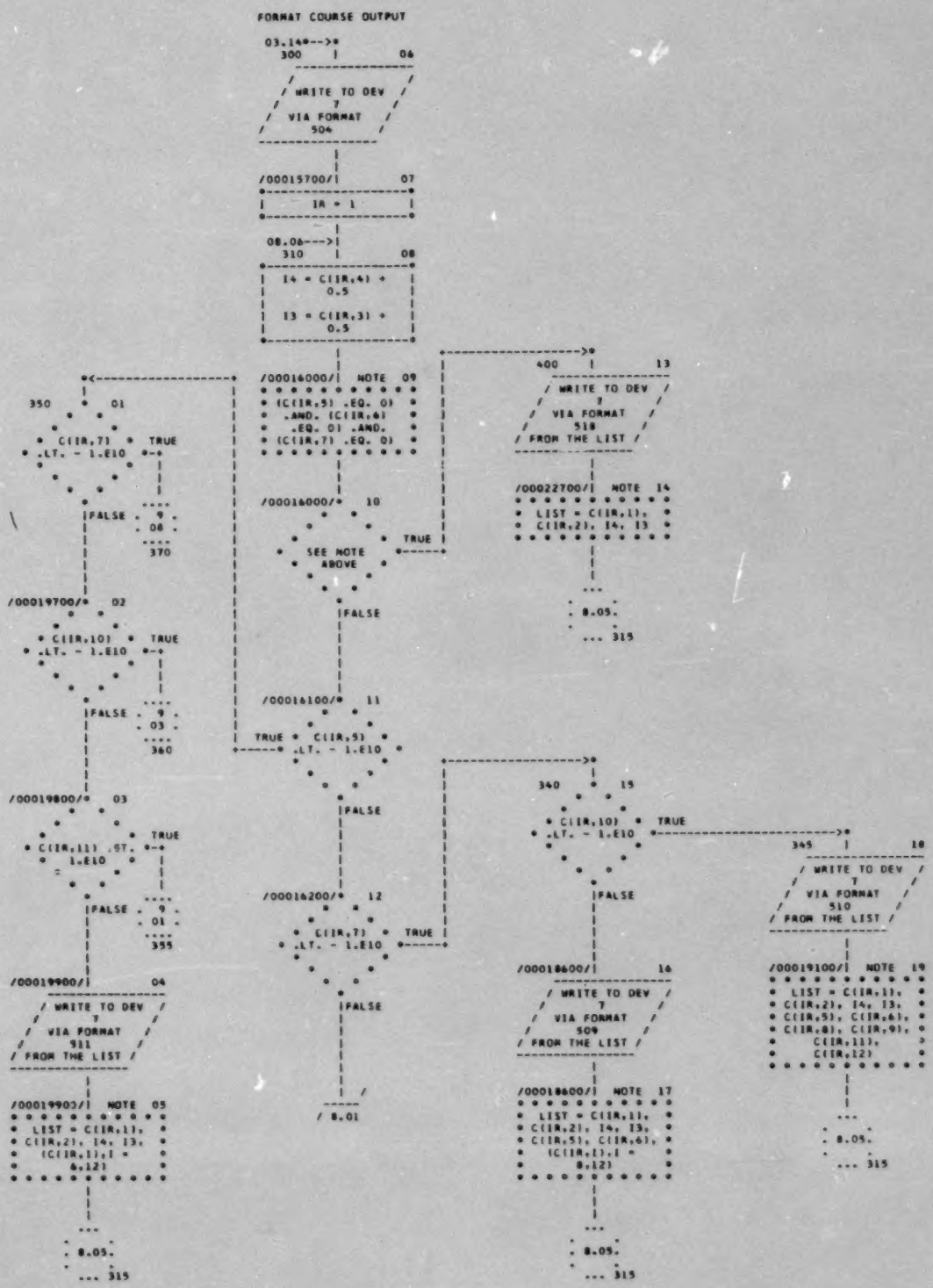


FIGURE IV-10 (CONTINUED) 413

CHART TITLE - SUBROUTINE FORMAT

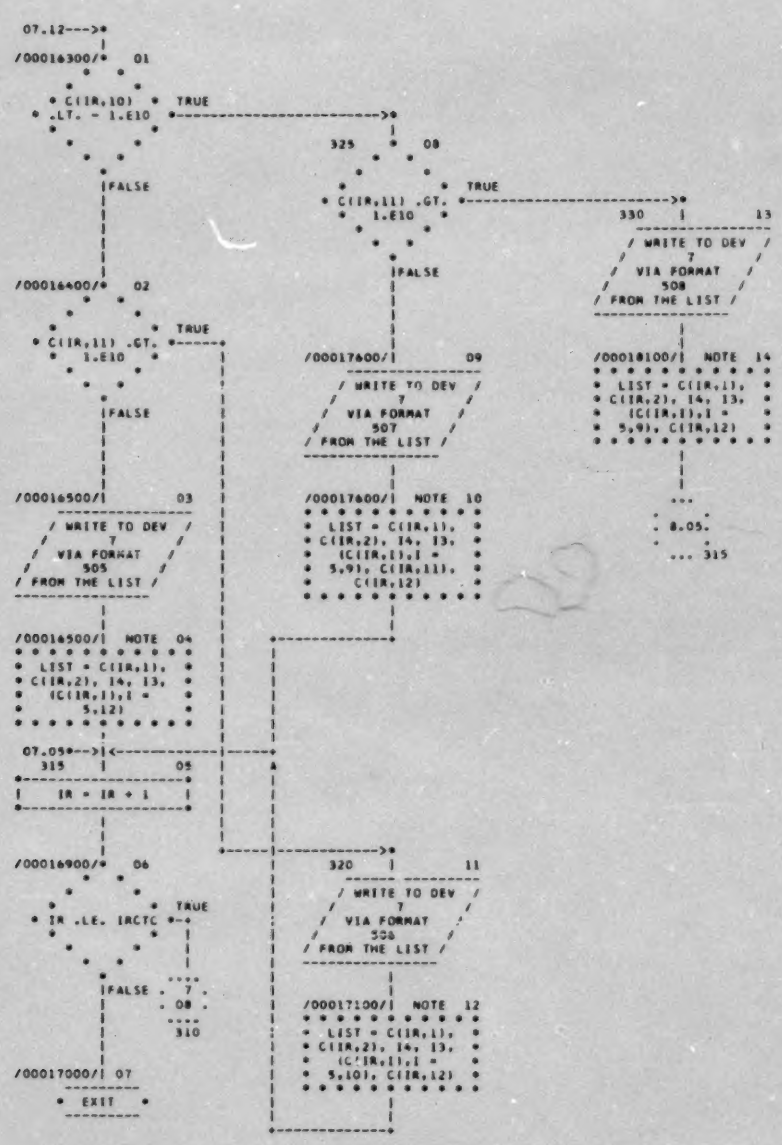


FIGURE IV-10 (CONTINUED)

CHART TITLE - SUBROUTINE FORMAT

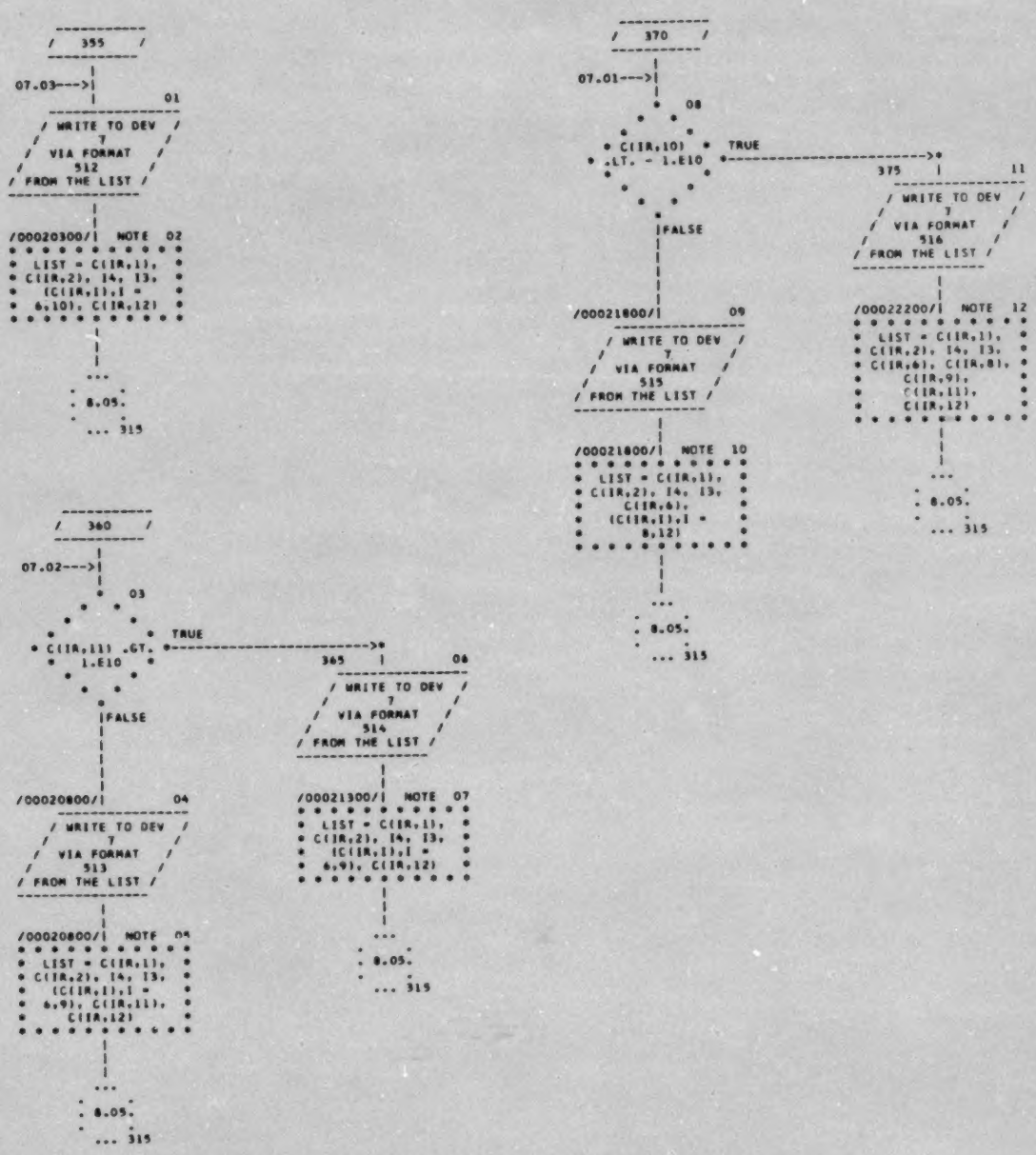


TABLE IV-10 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      REAL*8 A(9)/990.0/,NAME,R(500,10)/5000*0.0/,C(150,12)/1800*0.0/
      REAL*8 SEC1/'SECTION1'/,RSEC/'RSECTION'/,CSEC/'CSECTION'/
      REAL*8 SEC2/'SECTION2'/,SEC3/'SECTION3'/,SEC4/'SECTION4'/
      INTEGER FILE/4/
500   FORMAT (51X,'RESOURCE THROUGHPUT'/ 12X,'ANNUAL    ANNUAL    ',
      'HOURS     PERCENT   UTIL.     CHANGE   LIMITING'/1X,'RESOURCE',
      '   AVAIL.    UTIL.     UNDER',6X,'UTIL.    RANGE    PER UNIT  ',
      'VARIABLE'/ 12X,'(HRS)',5X,'(HRS)    UTILIZED',12X,'MINIMUM   ',
      'RESOURCES'/ 51X,'MAXIMUM    CHANGE')
501   FORMAT (1X,A8,2X,I6,4X,I6,4X,I6,6X,I4,3X,F8.1,3X,
      F7.3,3X,A8/ 50X,F8.1,3X,F7.3,3X,A8)
502   FORMAT (1X,A8,2X,I6,2(4X,I6),6X,I4,3X,F8.1,6X,'-INF',3X,A8/
      50X,F8.1,3X,F7.3,3X,A8)
503   FORMAT (1X,A8,2X,I6,2(4X,I6),6X,I4,3X,F8.1,3X,F7.3,3X,A8/
      50X,F8.1,6X,'-INF',3X,A8)
520   FORMAT (1X,A8,2X,I6,2(4X,I6),6X,I4,3X,F8.1,6X,'-INF',3X,A8/
      50X,F8.1,6X,'-INF',3X,A8)
517   FORMAT (1X,A8,2X,I4,2(4X,I6),6X,I4/' ')
504   FORMAT (' '/' '/' '/42X,'ANNUAL  THROUGHPUT  CLASS'/2X,'COURSE',
      '    MAXIMUM   CURRENT   AVERAGE   CONVEN    CHANGE     SIZE  ',
      '  LIMITING'/ 2X,'CDP NO.   ANNUAL   SCHEDULED   CLASS    RANGE',
      '   PER COURSE  RANGE    VARIABLE'/ 12X,'CONVEN    CONVEN   ',
      '  SIZE    MINIMUM    CONVEN   MINIMUM'/ 41X,'MAXIMUM    ',
      'CHANGE   MAXIMUM')
505   FORMAT (3X,A4,4X,F6.1,2(6X,I4),3(4X,F6.1),4X,A8/
      37X,3(4X,F6.1),4X,A8)
506   FORMAT (3X,A4,4X,F6.1,2(6X,I4),3(4X,F6.1),4X,A8/
      37X,2(4X,F6.1),6X,'+INF',4X,A8)
507   FORMAT (3X,A4,4X,F6.1,2(6X,I4),3(4X,F6.1),4X,A8/
      37X,4X,F6.1,6X,'-INF',4X,F6.1,4X,A8)
508   FORMAT (3X,A4,4X,F6.1,2(6X,I4),3(4X,F6.1),4X,A8/ 37X,4X,F6.1,
      6X,'-INF',6X,'+INF',4X,A8)
509   FORMAT (3X,A4,4X,F6.1,2(6X,I4),2(4X,F6.1),6X,'-INF',4X,A8/
      37X,3(4X,F6.1),4X,A8)
510   FORMAT (3X,A4,4X,F6.1,2(6X,I4),2(4X,F6.1),6X,'-INF',4X,A8/
      37X,4X,F6.1,6X,'+INF',4X,A8)
511   FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',2(4X,F6.1),4X,A8/
      37X,3(4X,F6.1),4X,A8)
512   FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',2(4X,F6.1),4X,A8/
      37X,2(4X,F6.1),6X,'+INF',4X,A8)
513   FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',2(4X,F6.1),4X,A8/
      37X,4X,F6.1,6X,'-INF',4X,F6.1,4X,A8)
514   FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',2(4X,F6.1),4X,A8/
      37X,4X,F6.1,6X,'-INF',6X,'+INF',4X,A8)
515   FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',4X,F6.1,6X,'-INF',
      4X,A8/ 37X,3(4X,F6.1),4X,A8)
516   FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',4X,F6.1,6X,'-INF',
      4X,A8/ 37X,4X,F6.1,6X,'-INF',4X,F6.1,4X,A8)
518   FORMAT (3X,A4,4X,F6.1,2(6X,I4)/' ')
519   FORMAT('*** READCOMM END OF FILE ERROR!!!!!!!')
```

FIGURE IV-10 (CONTINUED)

SCRR1 PROGRAM LISTINGS

```
SCRR1     EXEC
         VP/CSS

&TYPE OFF
ROUTE E REMOTE
ERASE MPSOUT LISTING
FILEDEF FT04F001 DSK FILE FT04F001
FILEDEF FT07F001 DSK SCRR OUTPUT RECFM UA BLKSIZE 132
&STACK KT
&STACK N
&BEGSTACK

&ENDSTACK
&STACK KT
&STACK MATRIX1
&BEGSTACK

FT04F001 FT07F001


Y
&ENDSTACK
&STACK RUN SCRR
ATTACH MPSIII ALL BRIEF
DETACH MPSIII ALL BRIEF
&PRI TO PRINT SCRR MODEL OUTPUT, TYPE:
&PRI          P SCRR OUTPUT
&EXIT
```

TABLE IV-12

INPUT LISTING AUTOFLOW CHART SET - FINAL -SCARFORM- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

```
CARD NO      ****                    CONTENTS                    ****

  1                SUBROUTINE FORMAT                                    00000100
  2                REAL*8 A(9)/9*0.0/,NAME,R(500,10)/5000*0.0/,C(150,12)/1800*0.0/   00000200
  3                REAL*8 SEC1/'SECTION1'/,RSEC/'RSECTION'/,CSEC/'CSECTION'/   00000300
  4                REAL*8 SEC2/'SECTION2'/,SEC3/'SECTION3'/,SEC4/'SECTION4'/   00000400
  5                INTEGER FILE/4/                                      00000500
  6          C                                                          00000600
  7          C GET RSECTION ARRAY                                       00000700
  8          C                                                          00000800
  9                CALL POSITN(FILE,1,2)                                00000900
 10                CALL ARRAY(FILE,I,NAME,N)                            00001000
 11                IF (I .EQ.1) GO TO 600                               00001100
 12                IR = 1                                               00001200
 13                IRCTR= 0                                             00001300
 14          5     CALL VECTOR (FILE,I,A)                               00001400
 15                IF (I .EQ. 1) GO TO 20                               00001500
 16                IRCTR = IRCTR + 1                                    00001600
 17                DO 10 I=1,4                                          00001700
 18          10    R(IR,I) = A(I)                                       00001800
 19                IR = IR + 1                                          00001900
 20                GO TO 5                                              00002000
 21          C                                                          00002100
 22          C  GET SECTION1 ARRAY                                      00002200
 23          C                                                          00002300
 24          20    CALL POSITN(FILE,1,5)                                00002400
 25                CALL ARRAY(FILE,I,NAME,N)                            00002500
 26                IF (I .EQ. 1) GO TO 200                              00002600
 27                IF (NAME .EQ. RSEC) GO TO 200                        00002700
 28                IF (NAME .NE. SEC1) GO TO 65                         00002800
 29          C                                                          00002900
 30          C ADD SENSITIVITY ANALYSIS DATA TO RESOURCE MATRIX         00003000
 31          C                                                          00003100
 32          25    IR = 1                                               00003200
 33          30    CALL VECTOR(FILE,I,A)                                00003300
 34                IF (I .EQ. 1) GO TO 60                               00003400
 35                DO 40 I=IR,IRCTR                                     00003500
 36                IF (A(1) .EQ. R(I,1)) GO TO 50                       00003600
 37          40    IR = IR + 1                                          00003700
 38          50    DO 55 I = 1,6                                        00003800
 39          55    R(IR,4+I) = A(1+I)                                   00003900
 40                IR = IR + 1                                          00004000
 41                GO TO 30                                             00004100
 42          60    IF (NAME .NE. SEC1) GO TO 200                        00004200
 43                IF (NAME .EQ. SEC3) GO TO 200                        00004300
 44          C                                                          00004400
 45          C GET SECTION3 ARRAY                                       00004500
 46          C                                                          00004600
 47          65    IA = 7                                               00004700
 48                IF (NAME .EQ. SEC1) GO TO 68                         00004800
 49                IA = 6                                               00004900
 50          68    CALL POSITN(FILE,1,IA)                               00005000
 51                CALL ARRAY(FILE,I,NAME,N)                            00005100
 52                IF (I .EQ. 1) GO TO 600                              00005200
 53                GO TO 25                                             00005300
 54          C                                                          00005400
 55          C GET CSECTION ARRAY                                       00005500
 56          C                                                          00005600
 57          70    CALL POSITN(FILE,1,3)                                00005700
 58                CALL ARRAY(FILE,I,NAME,N)                            00005800
 59                IF(I .EQ. 1) GO TO 600                               00005900
 60                IR = 1                                               00006000
 61                IRCTC = 0                                            00006100
 62          75    CALL VECTOR(FILE,I,A)                                00006200
 63                IF (I .EQ. 1) GO TO 90                               00006300
 64                IRCTC = IRCTC + 1                                    00006400
 65                DO 80 I =1,4                                         00006500
 66          80    C(IR,I) = A(I)                                       00006600
 67                IR = IR +1                                           00006700
 68                GO TO 75                                             00006800
 69          C                                                          00006900
 70          C GET SECTION2 ARRAY                                       00007000
 71          90    CALL POSITN(FILE,1,6)                                00007100
 72                CALL ARRAY(FILE,I,NAME,N)                            00007200
 73                IF (I .EQ. 1) GO TO 300                              00007300
 74                IF (NAME .EQ. CSEC) GO TO 300                        00007400
 75                IF (NAME .NE. SEC2) GO TO 130                        00007500
 76          C                                                          00007600
 77          C ADD SENSITIVITY ANALYSIS DATA TO COURSE MATRIX           00007700
 78          C                                                          00007800
 79          95    IR = 1                                               00007900
 80          100   CALL VECTOR(FILE,I,A)                               00008000
 81                IF(I .EQ. 1) GO TO 150                               00008100
 82                DO 110  I = IR,IRCTC                                 00008200
 83                IF (A(1) .EQ. C(I,1)) GO TO 120                      00008300
 84          110   IR = IR + 1                                          00008400
 85          120   DO 125 I =1,8                                        00008500
 86          125   C(IR,4+I) = A(1+I)                                   00008600
 87                IR = IR + 1                                          00008700
 88                GO TO 100                                            00008800
 89          130   CALL POSITN(FILE,1,5)                               00008900
 90                CALL ARRAY(FILE,I,NAME,N)                            00009000
 91                IF (I .EQ. 1) GO TO 600                              00009100
 92                IF (NAME .EQ. SEC2) GO TO 95                         00009200
 93          C                                                          00009300
 94          C GET SECTION4 ARRAY                                       00009400
 95          C                                                          00009500
 96          150   IF (NAME .EQ. SEC4) GO TO 300                       00009600
 97                CALL POSITN(FILE,1,6)                                00009700
 98                CALL ARRAY(FILE,I,NAME)                              00009800
 99                IF (NAME .EQ. SEC4) GO TO 95                         00009900
100                IF (NAME .EQ. SEC3) GO TO 300                        00010000
101                CALL POSITN(FILE,1,7)                                00010100
102                CALL ARRAY(FILE,I,NAME)                              00010200
103                IF (NAME .EQ. SEC4) GO TO 95                         00010300
104                CALL POSITN(FILE,1,8)                                00010400
105                CALL ARRAY(FILE,I,NAME)                              00010500
106                IF (I .EQ. 1) GO TO 300                              00010600
107                GO TO 95                                             00010700
108          C                                                          00010800
109          C FORMAT RESOURCE OUTPUT                                   00010900
```

TABLE IV-12 (CONTINUED)

418

INPUT LISTING — AUTOFLOW CHART SET - FINAL — -SCRAFORM- FLOWCHARTS

CARD NO — **** CONTENTS ****

```
C
 200  WRITE(7,500)
 500  FORMAT (51X,'RESOURCE THROUGHPUT'/ 12X,'ANNUAL     ANNUAL     ',
     1'HOURS      PERCENT   UTIL.      CHANGE    LIMITING'/1X,'RESOURCE',
     1'   AVAIL.     UTIL.      UNDER',6X,'UTIL.      RANGE      PER      ',
     1'VARIABLE'/ 12X,'(HRS)',5X,'(HRS)      UTILIZED',12X,'MINIMUM     ',
     1'RESOURCES'/ 51X,'MAXIMUM     CHANGE')
      IR = 2
 210  I4 = R(IR,4) + 0.5
      I2 = R(IR,2) + 0.5
      I3 = R(IR,3) + 0.5
      I5 = (R(IR,2) / R(IR,4)) * 100
      IF ((R(IR,5).EQ.0).AND.(R(IR,6).EQ.0).AND.(R(IR,7).EQ.0))GO TO 250
      IF (R(IR,6) .LT. -1.E10) GO TO 215
      IF (R(IR,9) .LT. -1.E10) GO TO 220
      WRITE(7,501) R(IR,1),I4,I2,I3,I5,(R(IR,I),I=5,10)
 501  FORMAT (1X,A8,2X,I6,4X,I6,4X,I6,6X,I4,3X,F8.1,3X,
     1F7.3,3X,A8/ 50X,F8.1,3X,F7.3,3X,A8)
 211  IR = IR + 1
      IF (IR .LE. IRCTR) GO TO 210
      GO TO 70
 215  IF (R(IR,9) .LT. -1.E10) GO TO 225
      WRITE(7,502) R(IR,1),I4,I2,I3,I5,R(IR,5),(R(IR,I),I=7,10)
 502  FORMAT (1X,A8,2X,I6,2(4X,I6),6X,I4,3X,F8.1,6X,'-INF',3X,A8/
     150X,F8.1,3X,F7.3,3X,A8)
      GO TO 211
 220  WRITE(7,503) R(IR,1),I4,I2,I3,I5,(R(IR,I),I=5,8),R(IR,10)
 503  FORMAT (1X,A8,2X,I6,2(4X,I6),6X,I4,3X,F8.1,3X,F7.3,3X,A8/
     150X,F8.1,6X,'-INF',3X,A8)
      GO TO 211
 225  WRITE (7,520) R(IR,1),I4,I2,I3,I5,R(IR,5),R(IR,7),R(IR,8),R(IR,10)
 520  FORMAT (1X,A8,2X,I6,2(4X,I6),6X,I4,3X,F8.1,6X,'-INF',3X,A8/
     150X,F8.1,6X,'-INF',3X,A8)
      GO TO 211
 250  WRITE (7,517) R(IR,1),I4,I2,I3,I5
 517  FORMAT (1X,A8,2X,I6,2(4X,I6),6X,I4/' ')
      GO TO 211
C
C FORMAT COURSE OUTPUT
C
 300  WRITE(7,504)
 504  FORMAT (' '/' '/' '/42X,'ANNUAL  THROUGHPUT  CLASS'/2X,'COURSE',
     1'    MAXIMUM   CURRENT   AVERAGE   CONVEN    CHANGE     SIZE  ',
     1'  LIMITING'/ 2X,'COP NO.   ANNUAL   SCHEDULED   CLASS    RANGE',
     1'   PER COURSE  RANGE    VARIABLE'/ 12X,'CONVEN    CONVEN   ',
     1'  SIZE    MINIMUM    CONVEN   MINIMUM'/ 41X,'MAXIMUM    ',
     1'CHANGE   MAXIMUM')
      IR=1
 310  I4 = C(IR,4) + 0.5
      I3 = C(IR,3) + 0.5
      IF((C(IR,5).EQ.0).AND.(C(IR,6).EQ.0).AND.(C(IR,7).EQ.0))GO TO 400
      IF (C(IR,5) .LT. -1.E10) GO TO 350
      IF (C(IR,7) .LT. -1.E10) GO TO 340
      IF (C(IR,10).LT. -1.E10) GO TO 325
      IF (C(IR,11).GT.  1.E10) GO TO 320
      WRITE(7,505) C(IR,1),C(IR,2),I4,I3,(C(IR,I),I=5,12)
 505  FORMAT (3X,A4,4X,F6.1,2(6X,I4),3(4X,F6.1),4X,A8/
     137X,3(4X,F6.1),4X,A8)
 315  IR=IR + 1
      IF (IR .LE. IRCTC) GO TO 310
      RETURN
 320  WRITE(7,506) C(IR,1),C(IR,2),I4,I3,(C(IR,I),I=5,10),C(IR,12)
 506  FORMAT (3X,A4,4X,F6.1,2(6X,I4),3(4X,F6.1),4X,A8/
     137X,2(4X,F6.1),6X,'+INF',4X,A8)
      GO TO 315
 325  IF (C(IR,11) .GT. 1.E10) GO TO 330
      WRITE(7,507) C(IR,1),C(IR,2),I4,I3,(C(IR,I),I=5,9),C(IR,11),
     1C(IR,12)
 507  FORMAT (3X,A4,4X,F6.1,2(6X,I4),3(4X,F6.1),4X,A8/
     137X,4X,F6.1,6X,'-INF',4X,F6.1,4X,A8)
      GO TO 315
 330  WRITE(7,508) C(IR,1),C(IR,2),I4,I3,(C(IR,I),I=5,9),C(IR,12)
 508  FORMAT (3X,A4,4X,F6.1,2(6X,I4),3(4X,F6.1),4X,A8/ 37X,4X,F6.1,
     16X,'-INF',6X,'+INF',4X,A8)
      GO TO 315
 340  IF (C(IR,10) .LT. -1.E10) GO TO 345
      WRITE(7,509) C(IR,1),C(IR,2),I4,I3,C(IR,5),C(IR,6),
     1(C(IR,I),I=8,12)
 509  FORMAT (3X,A4,4X,F6.1,2(6X,I4),2(4X,F6.1),6X,'-INF',4X,A8/
     137X,3(4X,F6.1),4X,A8)
      GO TO 315
 345  WRITE(7,510) C(IR,1),C(IR,2),I4,I3,C(IR,5),C(IR,6),C(IR,8),
     1C(IR,9),C(IR,11),C(IR,12)
 510  FORMAT (3X,A4,4X,F6.1,2(6X,I4),2(4X,F6.1),6X,'-INF',4X,A8/
     137X,4X,F6.1,6X,'+INF',4X,A8)
      GO TO 315
 350  IF (C(IR,7) .LT. -1.E10) GO TO 370
      IF (C(IR,10).LT. -1.E10) GO TO 360
      IF (C(IR,11).GT.  1.E10) GO TO 355
      WRITE(7,511) C(IR,1),C(IR,2),I4,I3,(C(IR,I),I=6,12)
 511  FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',2(4X,F6.1),4X,A8/
     137X,3(4X,F6.1),4X,A8)
      GO TO 315
 355  WRITE(7,512) C(IR,1),C(IR,2),I4,I3,(C(IR,I),I=6,10),C(IR,12)
 512  FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',2(4X,F6.1),4X,A8/
     137X,2(4X,F6.1),6X,'+INF',4X,A8)
      GO TO 315
 360  IF (C(IR,11) .GT. 1.E10) GO TO 365
      WRITE(7,513) C(IR,1),C(IR,2),I4,I3,(C(IR,I),I=6,9),C(IR,11),
     1C(IR,12)
 513  FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',2(4X,F6.1),4X,A8/
     137X,4X,F6.1,6X,'-INF',4X,F6.1,4X,A8)
      GO TO 315
 365  WRITE(7,514) C(IR,1),C(IR,2),I4,I3,(C(IR,I),I=6,9),C(IR,12)
 514  FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',2(4X,F6.1),4X,A8/
     137X,4X,F6.1,6X,'-INF',6X,'+INF',4X,A8)
      GO TO 315
 370  IF (C(IR,10) .LT. -1.E10) GO TO 375
      WRITE(7,515) C(IR,1),C(IR,2),I4,I3,C(IR,6),(C(IR,I),I=8,12)
 515  FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',4X,F6.1,6X,'-INF',
     16X,A8/ 37X,3(4X,F6.1),4X,A8)
      GO TO 315
```

419 TABLE IV-12 (CONTINUED)

INPUT LISTING AUTOFLOW CHART SET - FINAL -SCRRFORM- FLOWCHARTS

CARD NO **** CONTENTS ****

```
  375  WRITE(7,516) C(IR,1),C(IR,2),I4,I3,C(IR,6),C(IR,8),C(IR,9),
      1C(IR,11),C(IR,12)
  516  FORMAT (3X,A4,4X,F6.1,2(6X,I4),6X,'-INF',4X,F6.1,6X,'-INF',
      14X,A8/ 37X,4X,F6.1,6X,'-INF',4X,F6.1,4X,A8)
       GO TO 315
  400  WRITE (7,518) C(IR,1),C(IR,2),I4,I3
  518  FORMAT (3X,A4,4X,F6.1,2(6X,I4)/' ')
       GO TO 315
  600  WRITE (7,519)
  519  FORMAT('***  READCOMM  END OF FILE ERROR||||||')
       END
```

TABLE IV-12 (CONTINUED)

ASSIGN EXEC DESCRIPTION

The purpose of ASSIGN EXEC is to generate an instructor assignment listing which contains the following data for each instructor:

- o assigned courses
- o projected average class size
- o contact hour requirement per convening
- o scheduled convenings per year
- o total annual contact hour requirement per course
- o projected annual availability

Initial ASSIGN EXEC statements suppress all terminal output. Then DSRN's are associated with each of the input and output files as follows:

	DSRN	FILENAME	FILETYPE
Input	1	FILE	INST
	2	FILE	CRSE
Output	9	FILE	ASSIGN

After DSRN specification, ASSIGN1 FORTRAN is executed. ASSIGN1 FORTRAN output file FILE ASSIGN is then sorted by instructor using the NCSS SORT command. The sort key fields are 11-18. DSRN's are then reassigned as follows:

	DSRN	FILENAME	FILETYPE
Input	1	FILE	ASSIGN
	2	FILE	IAVAIL
Output	8	ASSIGN	LISTING

ASSIGN2 FORTRAN is then executed and instructions on how to obtain output are printed.

ASSIGN1 FORTRAN DESCRIPTION. The purpose of ASSIGN1 FORTRAN is to combine class size and convening data from FILE CRSE with each FILE INST record. FILE INST contains one record for each instructor assigned to a course. FILE CRSE contains one record for each course. Both files are sorted by CDP number.

ASSIGN1 reads the first record from both FILE INST and FILE CRSE. If both records contain the same CDP number, the data from both records are combined to form a single record which is written on FILE ASSIGN. The next record is then read from FILE INST. The CDP number is compared with the CDP number in the last record read from FILE CRSE. If they are identical, the data are combined and a single record is written on the output file. If the CDP numbers are not the same, the next record is read from FILE CRSE and the CDP comparison is repeated.

Records are read from both files, combined and rewritten on FILE ASSIGN until the last record is read from FILE INST.

ASSIGN2 FORTRAN DESCRIPTION. The purpose of ASSIGN2 FORTRAN is to combine instructor requirements and availability data and to format the data on the output file ASSIGN listing. Both input files FILE ASSIGN and FILE IAVAIL are sorted by instructor.

Upon entry, ASSIGN2 FORTRAN writes a heading on ASSIGN LISTING. Then the first record is read from FILE ASSIGN. Total contact hour requirements are calculated by multiplying the hours per convening by the number of convenings. The next record is read, total hours calculated, and the instructor name is compared with the instructor name of the prior record. If the names are different, availability is read from FILE IAVAIL, the former record is reformatted and written on ASSIGN LISTING, and the process is repeated.

If the names are identical, the first record is formatted and written on the output file, the next record read from FILE ASSIGN, and the instructor names are again compared. If the names are the same, the process is repeated. If the names are different, the former record is formatted and written on ASSIGN LISTING. Then availability is read from FILE IAVAIL, and a summary record, which includes total contact hours for all course assignments and total availability, is written on the output file.

Records are read from FILE ASSIGN, names compared, and output records written (format is a function of single vs. multiple assignments) until the last record is read from FILE ASSIGN.

ASSIGN EXEC FLOWCHART

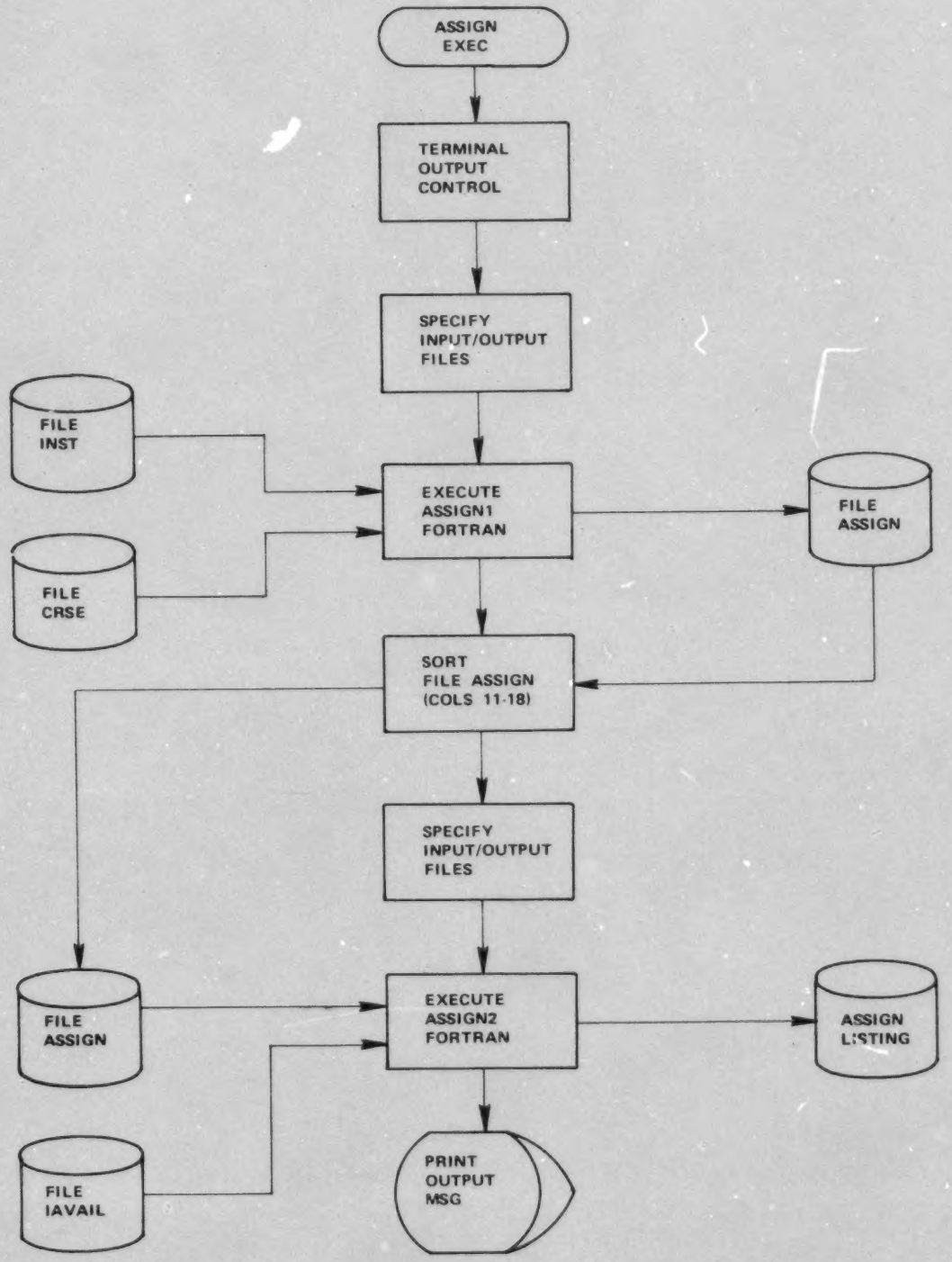
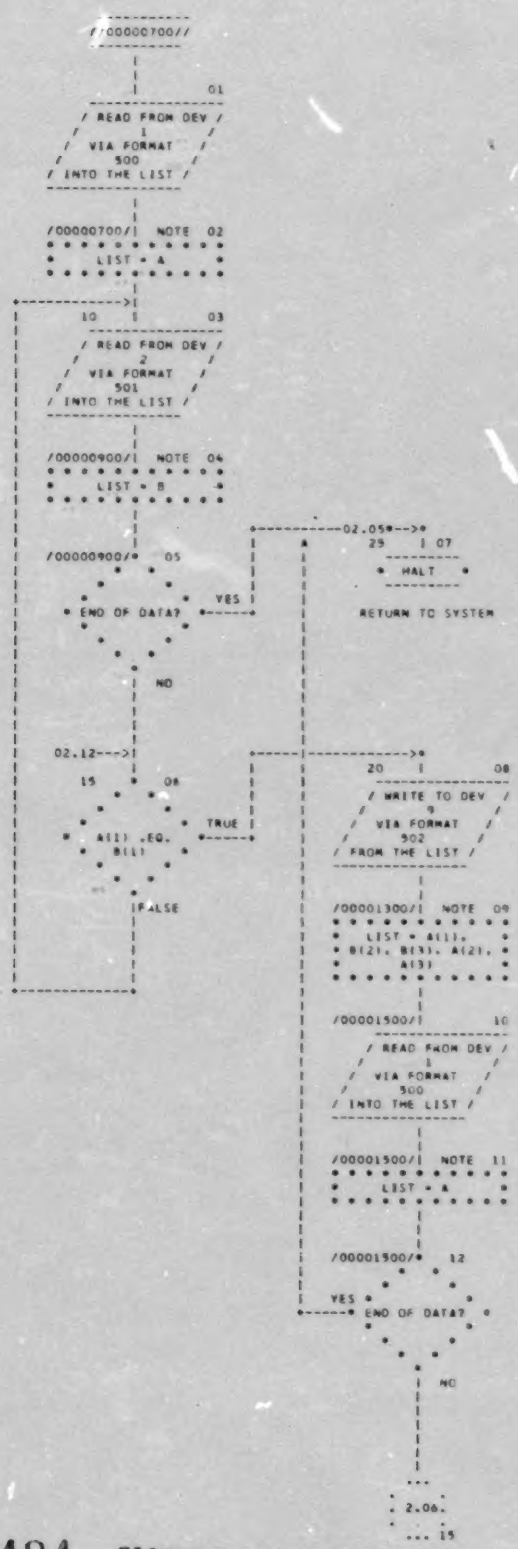


FIGURE IV-11

AUTOFLOW CHART SET - FINAL -ASSIGN1- FLOWCHARTS PAGE 12

CHART TITLE - PROCEDURES



424 FIGURE IV-11 (CONTINUED)

AUTOFLOW CHART SET - FINAL -ASSIGN1- FLOWCHARTS PAGE 03

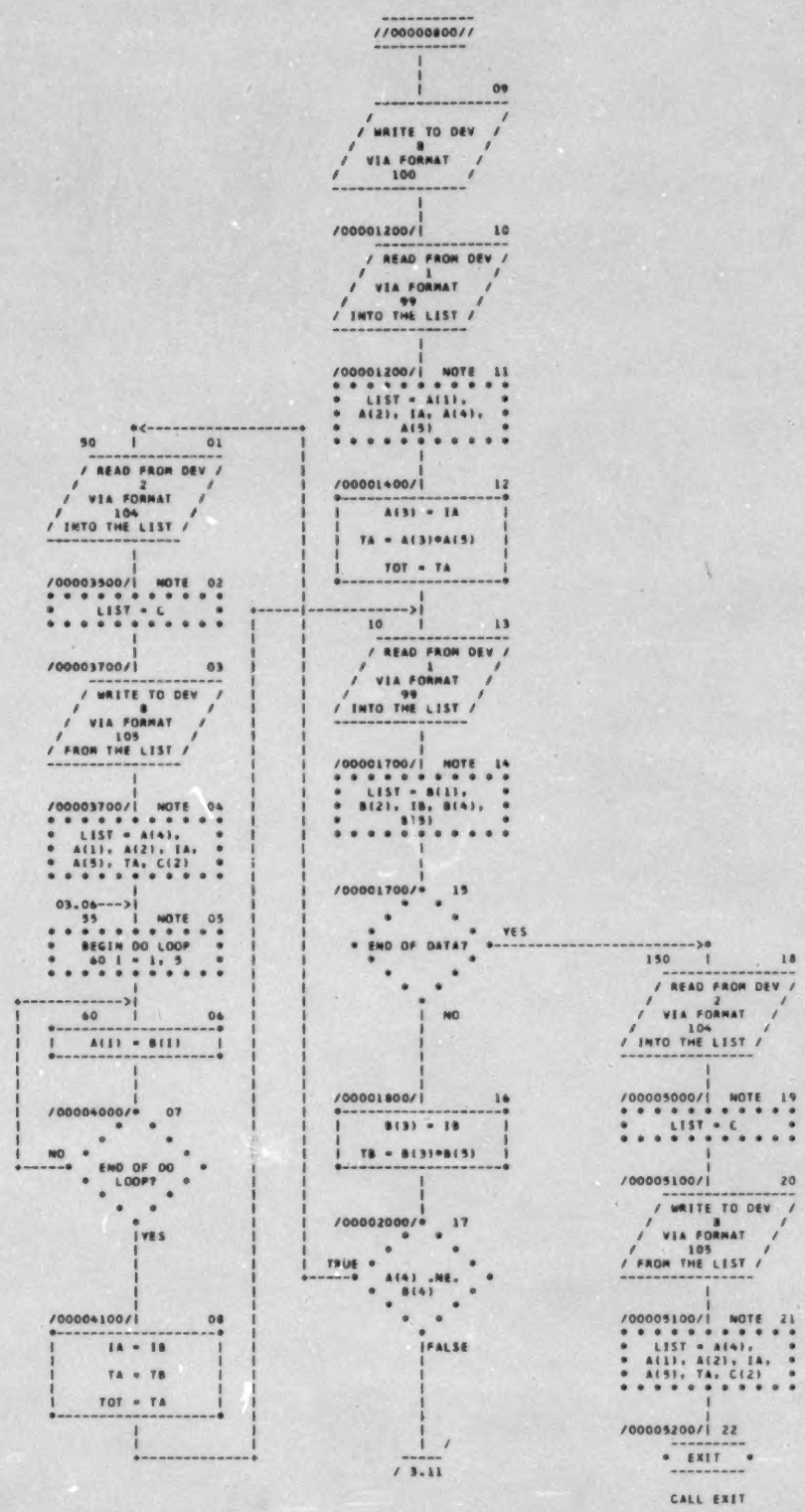
CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      REAL*8 A(3),B(3)
500   FORMAT (A4,A8,8X,F5.1)
501   FORMAT (A4,I3,I3)
502   FORMAT (A4,I3,I3,A8,F5.1)
```

FIGURE IV-11 (CONTINUED)

AUTOFLOW CHART SET - FINAL -ASSIGN2- FLOWCHARTS PAGE 02

CHART TITLE - PROCEDURES



426 FIGURE IV-11 (CONTINUED)

CHART TITLE - PROCEDURES

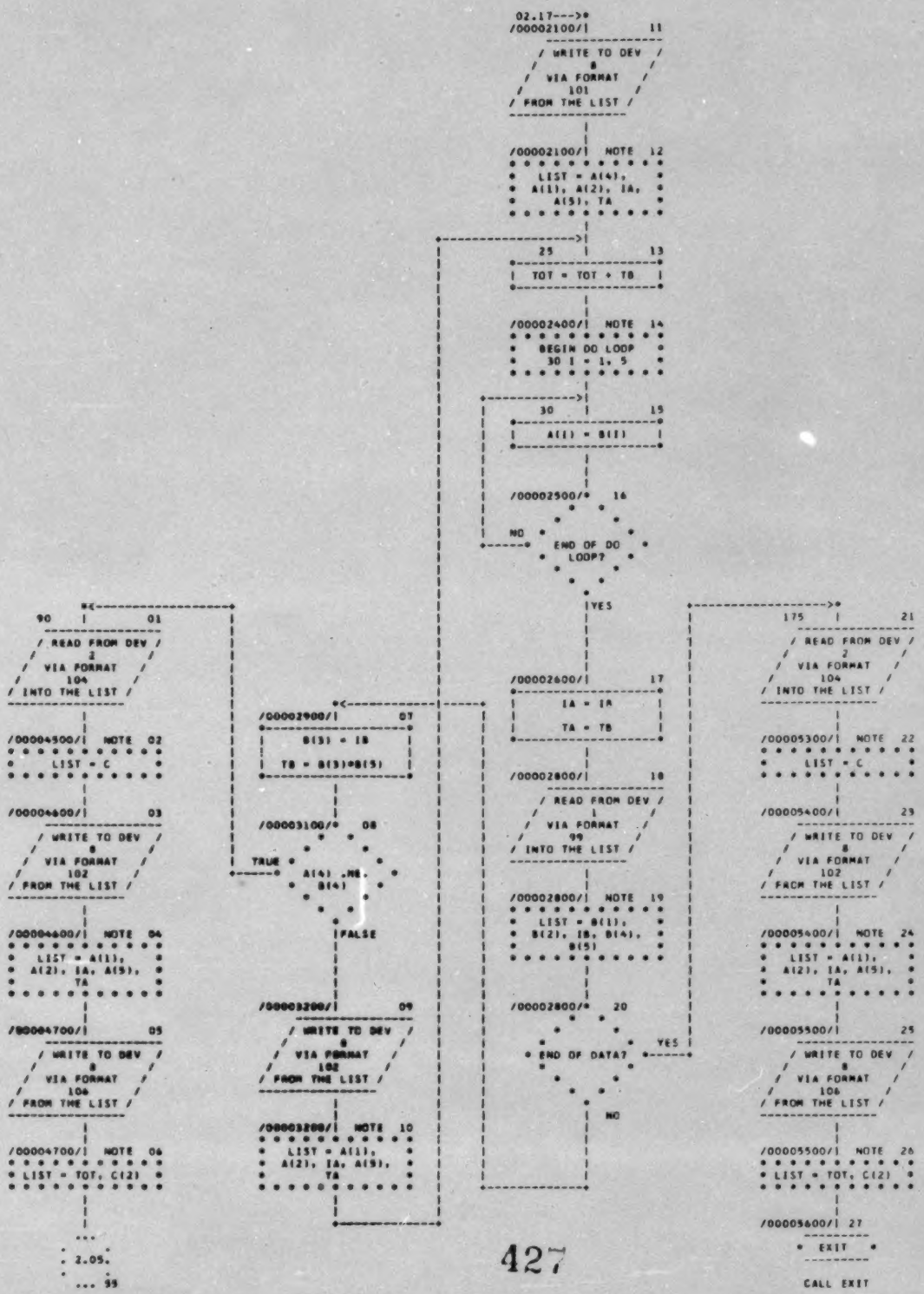


FIGURE IV-11 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      REAL*8 A(5),B(5),C(2)
100   FORMAT ('  INSTRUCTOR   COURSE  CLASIZE  CONV  HRS/CONV',
      '  TOT HRS  AVAIL HRS'/'  ----------   ------  -------  ',
      '----  --------  -------  ---------')
99    FORMAT (A4,I3,I3,A8,F5.1)
101   FORMAT (2X,A8,6X,A4,5X,I3,4X,I3,5X,F5.1,4X,F6.1)
102   FORMAT (16X,A4,5X,I3,4X,I3,5X,F5.1,4X,F6.1)
104   FORMAT (A8,8X,I4)
105   FORMAT (2X,A8,6X,A4,5X,I3,4X,I3,5X,F5.1,4X,F6.1,4X,I4/)
106   FORMAT (49X,'------'/43X,'TOT = ',F6.1,4X,I4/)
```

FIGURE IV-11 (CONTINUED)

ASSIGN PROGRAM LISTINGS

```
ASSIGN   EXEC
         VP/CSS

&TYPE OFF
&STACK KT
FILEDEF 1 DSK FILE INST
FILEDEF 2 DSK FILE CRSE
FILEDEF 9 DSK FILE ASSIGN
RUN ASSIGN1
&STACK 11 18
SORT FILE ASSIGN (BRIEF)
ERASE FILE ASSIGN
ALTER .FILE ASSIGN P FILE * *
FILEDEF 1 DSK FILE ASSIGN
FILEDEF 2 DSK FILE IAVAIL
FILEDEF 8 DSK ASSIGN LISTING
RUN ASSIGN2
&PRI TO PRINT INSTRUCTOR ASSIGNMENTS, TYPE:
&PRI              P ASSIGN LISTING
```

TABLE IV-13

INPUT LISTING AUTOFLOW CHART SET - FINAL -ASSIGN1- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
      REAL*8 A(3),B(3)
C**COMBINES INST AND CRSE DATA INTO SINGLE FILE
C  FILE 1 = FILE INST
C  FILE 2 = FILE CRSE
C  FILE 9 = FILE ASSIGN
C
      READ (1,500) A
  500 FORMAT (A4,A8,8X,F5.1)
   10 READ (2,501, END = 25) B
  501 FORMAT (A4,I3,I3)
   15 IF (A(1) .EQ. B(1) ) GO TO 20
      GO TO 10
   20 WRITE (9,502) A(1),B(2),B(3),A(2),A(3)
  502 FORMAT (A4,I3,I3,A8,F5.1)
      READ (1,500, END = 25) A
      GO TO 15
   25 STOP
      END
```

INPUT LISTING AUTOFLOW CHART SET - FINAL -ASSIGN2- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
      REAL*8 A(5),B(5),C(2)
C
C**FORMATS INST ASSIGNMENT DATA FOR OUTPUT
C  FILE 1 = FILE ASSIGN
C  FILE 2 = FILE IAVAIL
C  FILE 8 = ASSIGN LISTING
C
      WRITE (8,100)
  100 FORMAT ('  INSTRUCTOR   COURSE  CLASIZE  CONV  HRS/CONV',
     1'  TOT HRS  AVAIL HRS'/'  ----------   ------  -------  ',
     1'----  --------  -------  ---------')
      READ (1,99) A(1),A(2),IA,A(4),A(5)
   99 FORMAT (A4,I3,I3,A8,F5.1)
      A(3) = IA
      TA = A(3) * A(5)
      TOT = TA
   10 READ (1,99, END = 150) B(1),B(2),IB,B(4),B(5)
      B(3) = IB
      TB = B(3) * B(5)
      IF ( A(4) .NE. B(4) ) GO TO 50
      WRITE (8,101) A(4),A(1),A(2),IA,A(5),TA
  101 FORMAT (2X,A8,6X,A4,5X,I3,4X,I3,5X,F5.1,4X,F6.1)
   25 TOT = TOT + TB
      DO 30  I=1,5
   30 A(I) = B(I)
      IA = IB
      TA = TB
      READ (1,99, END = 175) B(1),B(2),IB,B(4),B(5)
      B(3) = IB
      TB = B(3) * B(5)
      IF ( A(4) .NE. B(4) ) GO TO 90
      WRITE (8,102) A(1),A(2),IA,A(5),TA
  102 FORMAT (16X,A4,5X,I3,4X,I3,5X,F5.1,4X,F6.1)
      GO TO 25
   50 READ (2,104) C
  104 FORMAT (A8,8X,I4)
      WRITE (8,105) A(4),A(1),A(2),IA,A(5),TA,C(2)
  105 FORMAT (2X,A8,6X,A4,5X,I3,4X,I3,5X,F5.1,4X,F6.1,4X,I4/)
   55 DO 60  I=1,5
   60 A(I) = B(I)
      IA = IB
      TA = TB
      TOT = TA
      GO TO 10
   90 READ (2,104) C
      WRITE (8,102) A(1),A(2),IA,A(5),TA
      WRITE (8,106) TOT,C(2)
  106 FORMAT (49X,'------'/43X,'TOT = ',F6.1,4X,I4/)
      GO TO 55
  150 READ (2,104) C
      WRITE (8,105) A(4),A(1),A(2),IA,A(5),TA,C(2)
      CALL EXIT
  175 READ (2,104) C
      WRITE (8,102) A(1),A(2),IA,A(5),TA
      WRITE (8,106) TOT,C(2)
      CALL EXIT
      END
```

TABLE IV-13 (CONTINUED)

MODAVAIL EXEC DESCRIPTION

The purpose of MODAVAIL EXEC is to modify the number of available hours associated with either an instructor or a facility in the LP input data file MATRIX1 DATA. MODAVAIL uses the CSS EDIT command to effect the file change. The EDIT command copies the MATRIX1 DATA file into core. Subsequent edit requests examine and modify the core image of the file and then write the file to disk.

The initial EXEC statement suppresses printing of all CSS command statements and all EXEC control statements. Next the output file for the FORTRAN program MODIFY is specified and MODIFY is executed. The EXEC file CHG which has been created by MODIFY is then executed. CHG EXEC uses the CSS EDIT command to modify the file MATRIX1 DITTA. First, the line containing the specified instructor or facility name is located in the file by the LOCATE request. Then the available hour figure contained in the file is replaced with a new figure using the EDIT command CHANGE request. An error in data entry will result in a NO CHANGE message, which indicates that MATRIX1 DATA has not been changed. The final statement in the CHG EXEC writes the updated MATRIX1 DATA file to a writable disk. CHG EXEC is erased following execution. If the user indicates that additional availability modifications are required, MODAVAIL EXEC branches to statement two and reexecutes MODIFY and CHG EXEC. If further changes are not required, a message informing the user how to rerun the SCRR model is printed.

MODIFY FORTRAN DESCRIPTION. The purpose of MODIFY FORTRAN is to create the CHG EXEC file containing the user-specified availability information.

The user specifies the name of the instructor of facility, current available hours, and the new available hours. This information is formatted within several CSS EDIT command request statements and written on output file CHG EXEC. The CHG EXEC file created by MODIFY is shown below.

```
CHG EXEC

&STACK KT
&BEGSTACK
L. /RHS/
L  /NAME/
C  /NAME            CHRS/NAME          NHRS/
FILE
&ENDSTACK
EDIT MATRIX1 DATA
```

MODAVAIL EXEC FLOWCHART

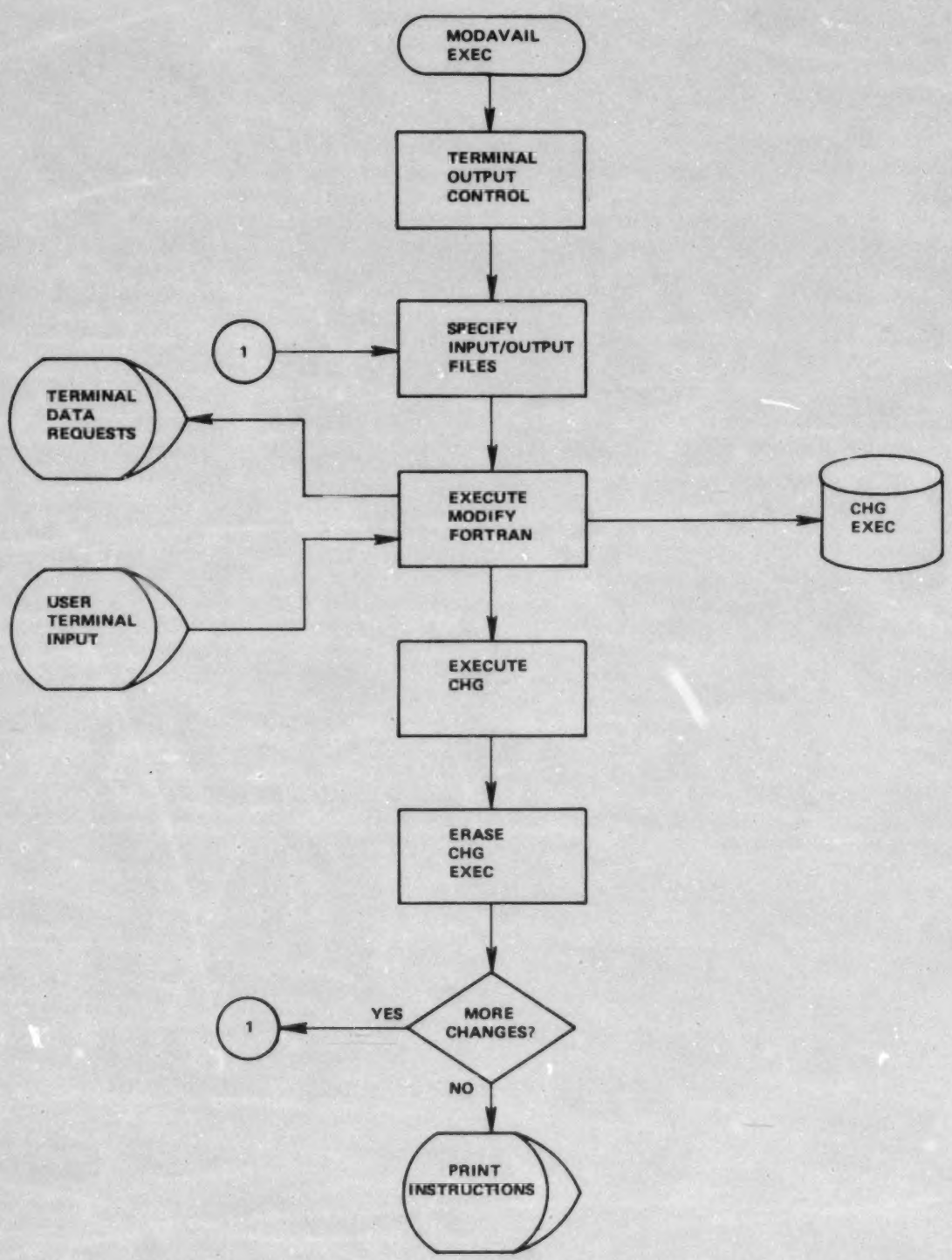


FIGURE IV-12

CHART TITLE - PROCEDURES

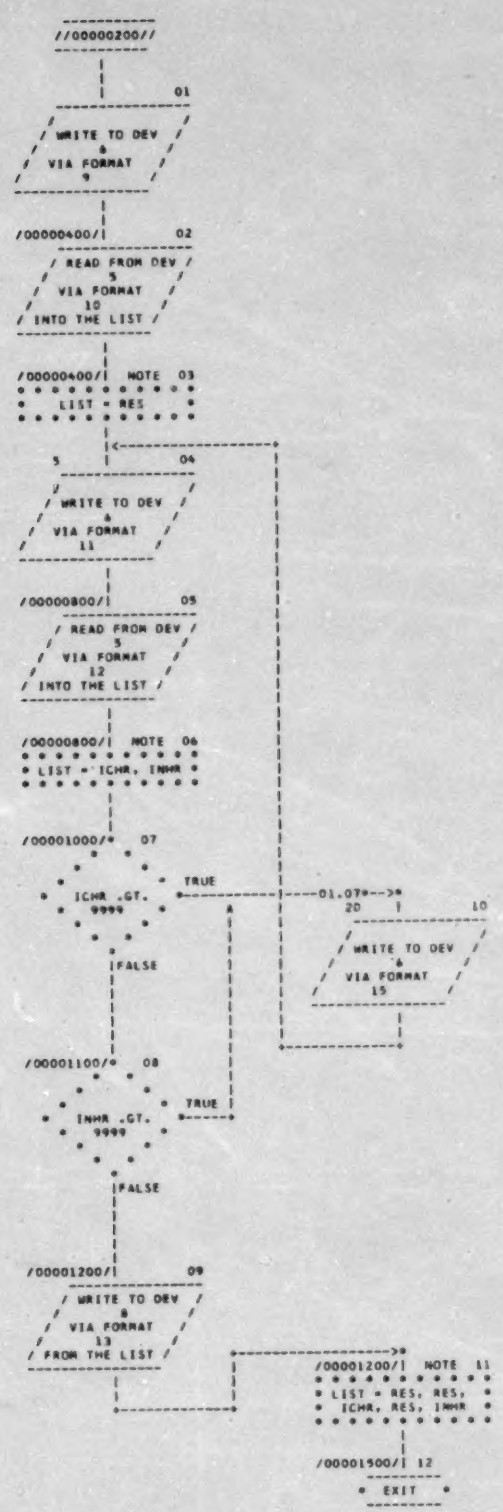


FIGURE IV-12 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      REAL*8 RES
9     FORMAT ('ENTER NAME OF INSTRUCTOR OR FACILITY: ')
10    FORMAT (A8)
11    FORMAT ('ENTER CURRENT AVAILABLE HOURS, NEW AVAILABLE HOURS: ')
12    FORMAT (2I99)
13    FORMAT ('&STACK KT'/'&BEGSTACK'/'L. /RHS/'/'L /',A8,'/'/'C /',A8,
      8X,I4,'/',A8,8X,I4,'/'/'FILE'/'&ENDSTACK'/'EDIT MATRIXL DATA')
15    FORMAT ('AVAILABLE HOURS LIMITED TO FOUR (4) DIGIT NUMBER  TRY ',
      'AGAIN.')
```

FIGURE IV-12 (CONTINUED)

MODAVAIL PROGRAM LISTINGS

```
MODAVAIL EXEC
        VP/CSS

&TYPE OFF
-CHG FILEDEF 8 DSK CHG EXEC
FILEDEF 6 CONO
RUN MODIFY
EXEC CHG
CSS ERASE CHG EXEC
&PRI ANY MORE CHANGES? (YES/NO)
&READ ARGS
&IF &1 NE NO &GOTO -CHG
&PRI CHANGES TO AVAILABLE HOURS HAVE BEEN INCORPORATED.
&PRI TO RERUN SCRR MODEL  TYPE:  SCRR1
&EXIT
```

FIGURE IV-14

INPUT LISTING AUTOFLOW CHART SET FINAL -MODIFY- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
      REAL*8 RES                                                        00000100
      WRITE(6,9)                                                        00000200
    9 FORMAT ('ENTER NAME OF INSTRUCTOR OR FACILITY: ')                 00000300
      READ (5,10) RES                                                   00000400
   10 FORMAT (A8)                                                       00000500
    5 WRITE (6,11)                                                      00000600
   11 FORMAT ('ENTER CURRENT AVAILABLE HOURS, NEW AVAILABLE HOURS: ')   00000700
      READ (5,12) ICHR,INHR                                             00000800
   12 FORMAT (2I99)                                                     00000900
      IF (ICHR .GT. 9999) GO TO 20                                      00001000
      IF (INHR .GT. 9999) GO TO 20                                      00001100
      WRITE (8,13) RES,RES,ICHR,RES,INHR                                00001200
   13 FORMAT ('&STACK KT'/'&BEGSTACK'/'L. /RHS/'/'L /',A8,'/'/'C /',A8, 00001300
     18X,I4,'/',A8,8X,I4,'/'/'FILE'/'&ENDSTACK'/'EDIT MATRIX1 DATA')    00001400
      CALL EXIT                                                         00001500
   20 WRITE (6,15)                                                      00001600
   15 FORMAT ('AVAILABLE HOURS LIMITED TO FOUR (4) DIGIT NUMBER  TRY ', 00001700
     1'AGAIN.')                                                         00001800
      GO TO 5                                                           00001900
      END                                                               00002000
```

TABLE IV-14 (CONTINUED)

REQMT EXEC DESCRIPTION

The purpose of REQMT EXEC is to provide the model user with the capability to modify course utilization, class capacity, and course length expressed in contact hours to assess the impact of these changes on total instructor and facility requirements. The EXEC utilizes three RAMIS RUN EXECUTIVE procedures to retrieve data from two RAMIS data files MODCOURSE and MODFACIL. The RUN EXECs extract current utilization, capacity and length data from the data files and then factor these data by the user-specified percentage changes prior to calculating and filing instructor and facility requirements. RUN EXECUTIVES are cataloged RAMIS procedures capable of accepting variable parameter input values.

The initial REQMT EXEC statement suppresses printing of all CSS command statements and all EXEC control statements. The user is then asked if program operating instructions should be printed. If the response is positive, several lines of text are printed prior to each data request. A negative response will cause all text statements to be bypassed. Values are then requested for the utilization percent change, capacity change, and the length change.

After the requested data have been entered, output files are specified, RAMIS is attached, and three RAMIS RUN EXECUTIVES are executed. REQMT uses all three parameters (utilization, capacity, and length) and generates the file FILE INST which contains instructor contact hour requirements by instructor by course. REQMT1 uses only the utilization and capacity parameters and generates the file FILE CRSE which contains the class size and number of convenings by course. REQMT2 uses the course length parameter and generates the file FACIL which contains the contact hours requirements by facility by course. After REQMT2 has been executed, RAMIS is detached and a message which indicates that the required output files have been generated is printed.

REQMT RUNEXEC DESCRIPTION. The purpose of the REQMT RUNEXEC is to calculate instructor requirements in hours by course and by instructor incorporating the course parameter deltas as specified by the user through the REQMT EXEC. The basic data required to determine instructor requirements are stored in the RAMIS course file MODCOURSE.

REQMT RUNEXEC is based on the RAMIS procedure SCRR INST described earlier. The primary difference between the two procedures is that the RUNEXEC factors the data retrieved from the course file by the user-specified percentages. REQMT output is written on FILE INST.

REQMT RUNEXEC is a cataloged procedure stored in the RAMIS file RAMEXEC. Maintenance of the RAMEXEC file to change a procedure can be performed in the UPDATE mode or in the SCAN mode through the usual RAMIS facilities. Alternately the file REQMT RUNEXEC can be modified, and the procedure recataloged by using the RAMIS command RAMISIN REQMT RUNEXEC.

REQMT1 RUNEXEC DESCRIPTION. The purpose of the REQMT1 RUNEXEC is to record the convening frequency and average class size for each course incorporating the user-specified course parameter deltas. The required ata are stored in the RAMIS course file MODCOURSE.

REQMT1 RUNEXEC is based on the RAMIS procedure SCRR CRSE described earlier. The primary difference between the two procedures is that the RUNEXEC factors the data retrieved from the course file by the user-specified percentages. REQMT1 output is written on FILE CRSE.

REQMT2 RUNEXEC DESCRIPTION. The purpose of the REQMT2 RUNEXEC is to determine facility requirements in hours by course and by facility incorporating the course length delta specified by the user. The basic data are stored in the RAMIS facility file MODFACIL.

REQMT2 RUNEXEC is based on the RAMIS procedure SCRR FACIL described earlier. The primary difference between the two procedures is that the RUN EXEC factors the data retrieved from the facility file by the user-specified percentages. REQMT2 output is written on FILE FACIL.

REQMT EXEC FLOWCHART

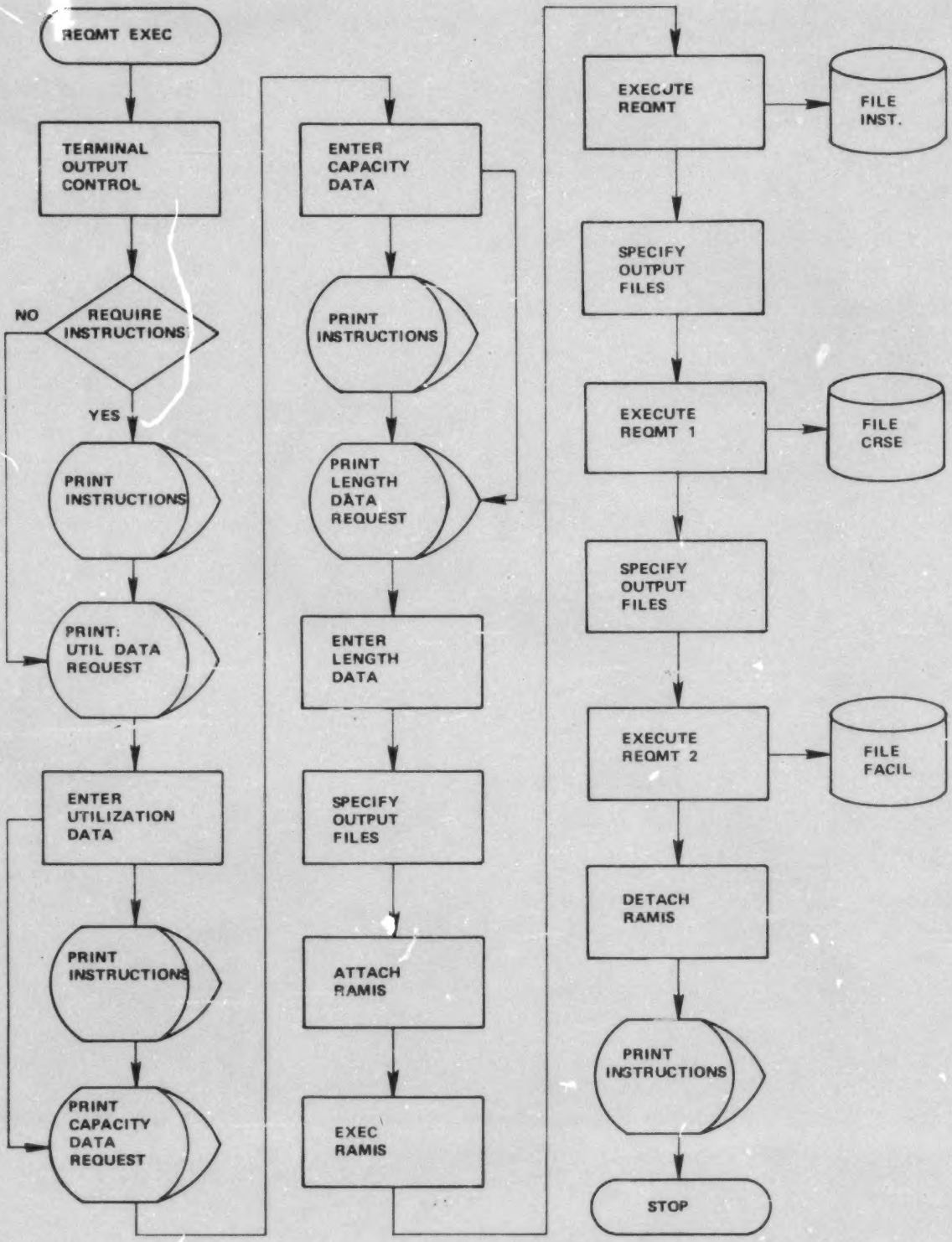


FIGURE IV-13

REQMT PROGRAM LISTINGS

```
REQMT    EXEC
         VP/CSS


&TYPE OFF
&PRINT  DO YOU WISH INSTRUCTIONS TO BE PRINTED? (YES/NO) :
&READ ARGS
&DIR = &1
&IF &DIR NE YES  &GOTO -U
&PRINT  THIS PROGRAM PROVIDES THE CAPABILITY TO MODIFY COURSE
&PRINT  UTILIZATION, CLASS CAPACITY, AND COURSE LENGTH EXPRESSED
&PRINT  IN CONTACT HOURS TO ASSESS THE IMPACT ON TOTAL INSTRUCTOR
&PRINT  REQUIREMENTS.
&PRINT   * TO INCREASE UTILIZATION 20% ENTER:   .20
&PRINT   * TO DECREASE UTILIZATION 10% ENTER:  -.10
&PRINT   * IF NO CHANGE IS DESIRED     ENTER:    0
-U  &PRINT   ENTER UTILIZATION DELTA:
&READ ARGS
&UDEL = &1
&IF &DIR NE YES  &GOTO -C
&PRINT   * TO INCREASE CLASS CAPACITIES 20% ENTER:   .20
&PRINT   * TO DECREASE CLASS CAPACITIES 10% ENTER:  -.10
&PRINT   * IF NO CHANGE IS DESIRED          ENTER:    0
-C  &PRINT   ENTER CLASS CAPACITY DELTA:
&READ ARGS
&CDEL = &1
&IF &DIR NE YES &GOTO -L
&PRINT  * TO INCREASE COURSE LENGTH 20% ENTER:   .20
&PRINT  * TO DECREASE COURSE LENGTH 10% ENTER:  -.10
&PRINT  * IF NO CHANGE IS DESIRED       ENTER:    0
-L  &PRINT   ENTER COURSE LENGTH DELTA:
&READ ARGS
&LDEL = &1
&STACK KT
&STACK EXEC REQMT &UDEL , &CDEL , &LDEL
FILEDEF 6 MSG KILL
FILEDEF 4 DSK FILE INST RE F LR 25
ATTACH RAMIS ALL BRIEF
EXEC RAMIS
DETACH RAMIS ALL BRIEF
&STACK KT
&STACK EXEC REQMT1 &UDEL , &CDEL
FILEDEF 4 DSK FILE CRSE RE F LR 10
ATTACH RAMIS ALL BRIEF
EXEC  RAMIS
DETACH RAMIS ALL BRIEF
&STACK KT
&STACK EXEC REQMT2 &LDEL
&STACK FIN
FILEDEF 4 DSK FILE FACIL RE F LR 22
ATTACH RAMIS ALL BRIEF
EXEC RAMIS
DETACH RAMIS ALL BRIEF
ERASE MSG KILL
FILEDEF 6 CONO
&PRI REQUESTED CHANGES HAVE BEEN INCORPORATED.
&PRI TO RERUN SCRR MODEL  TYPE:  SCRR2
```

TABLE IV-15

```
REQMT    RUNEXEC
        VP/CSS


DELETE
FILE RAMEXEC
1=OUT=REQMT,$
END
CATALOG REQMT
DEFINE
FILE MODCOURSE
UCAP1 / F4.2 = MIN (1.0, (UCAP * .01) + (UCAP * .01 * (&1)))
CLASIZE / I3 = (CAP1 * UCAP1 * ((&2) + 1.0)) + .5
NOINST / I3 = MAX (1.0, (CLASIZE / SIR) + .5)
TOTHRS / F6.1 = NOINST * CHRS * (&3 + 1.0)
FLAG / I1 = ASN
END
TABLE
FILE MODCOURSE
STORE TOTHRS AND FLAG BY CDP BY SIR BY NME
IF DEL IS 1
RUN
FILE MODCOURSE
SUM ASN AND HOLD BY CDP BY SIR
END
DEFINE
FILE HOLD
INSTHRS / F5.1 = TOTHRS / ASN
END
TABLE
FILE HOLD
SUM INSTHRS AND SAVE BY CDP BY NME
IF FLAG IS-MORE-THAN 0
IF NME IS-NOT '                    '
END
FIN
END CATALOG
```

TABLE IV-15 (CONTINUED)

```
REQMT1   RUNEXEC
         VP/CSS

DELETE
FILE RAMEXEC
1=OUT=REQMT1.$
END
CATALOG REQMT1
DEFINE
FILE MODCOURSE
UCAP1 / F4.2 = MIN (1.0, (.01 * UCAP) + (UCAP * .01 * (&1)))
CLASIZE / I3 = (CAP1 * UCAP1 * ((&2) + 1.0)) + .5
END
TABLE
FILE MODCOURSE
PRINT CLASIZE AND CONV1 AND SAVE BY CDP
IF DEL IS 1
END
FIN
END CATALOG
```

```
REQMT2   RUNEXEC
         VP/CSS

DELETE
FILE RAMEXEC
1=OUT=REQMT2.$
END
CATALOG REQMT2
DEFINE
FILE MODFACIL
RHRS / F6.1 = REQHRS + (REQHRS * (&1))
END
TABLE
FILE MODFACIL
PRINT RHRS AND SAVE BY CDP BY BLDG BY RM
END
FIN
END CATALOG
```

TABLE IV-15 (CONTINUED)

SCRR2 EXEC DESCRIPTION

The purpose of the SCRR2 EXEC is to incorporate the resource requirement changes requested through the REQMT EXEC into the LP input data matrix and then rerun the SCRR model.

REQMT EXEC regenerates three RAMIS interface files - FILE INST, FILE FACIL, and FILE CRSE. To incorporate the data from these files into the LP data matrix, it is necessary to execute those executives which require any of these files as input. Therefore, the following executives are serially executed by the SCRR2 EXEC:

BLANK
IF2
SCRRSORT
COMB
SCRR1

SCRR2 PROGRAM LISTING

```
SCRR2    EXEC
         VP/CSS

&PRI THIS EXECUTIVE WILL INCORPORATE THE CHANGES
&PRI REQUESTED BY THE REQMT EXECUTIVE INTO THE LP
&PRI INPUT AND RERUN THE SCRR MODEL
&TYPE OFF
EXEC BLANK
EXEC IF2
EXEC SCRRSORT
EXEC COMB
EXEC SCRR1
```

TABLE IV-16

FIXCONV EXEC DESCRIPTION

The purpose of the FIXCONV EXEC is to modify the LP input data MATRIX1 DATA to fix the number of course convenings at the current level.

The MATRIX1 DATA file contains one BOUND data card image for each course in the current problem formulation. Columns 2-3 of this card image contain the bound type indicator. The indicator LO indicates that the number of convenings specified in columns 25-36 is a lower bound. This is equivalent to adding a constraint to the LP problem formulation specifying that the number of convenings must be greater than or equal to the number specified on the BOUND data card. FIXCONV EXEC uses the NCSS EDIT command to change the BOUND type indicator LO on all BOUND data cards in MATRIX1 DATA to the type indicator FX. FX indicates that the number of convenings specified on the BOUND data card is a fixed value.

FIXCONV PROGRAM LISTING

```
FIXCONV  EXEC
         VP/CSS

&BEGSTACK
KT
C /LO/FX/ * *
FILE
&ENDSTACK
EDIT MATRIX1 DATA SB
&PRI COURSE CONVENINGS FIXED AT THE CURRENT LEVEL.
&PRI TO RERUN SCRR MODEL TYPE : SCRR1
&EXIT
```

TABLE IV-17

MAXCONV EXEC DESCRIPTION

The purpose of the MAXCONV EXEC is to modify the LP input data MATRIX1 DATA to remove the upper bound on course convenings previously established through the execution of FIXCONV EXEC.

MAXCONV EXEC reverses the data change made by FIXCONV EXEC. MAXCONV uses the NCSS EDIT command to change the BOUND type indicator FX on all BOUND data cards on MATRIX1 DATA to the type indicator LO.

If the type indicator is not FX when MAXCONV is executed, the program will respond with the message NO CHANGE.

MAXCONV PROGRAM LISTING

```
MAXCONV  EXEC
         VP/CSS

&BEGSTACK
KT
C /FX/LO/ * *
FILE
&ENDSTACK
EDIT MATRIX1 DATA SB
```

TABLE IV-18

(THIS PAGE INTENTIONALLY LEFT BLANK)

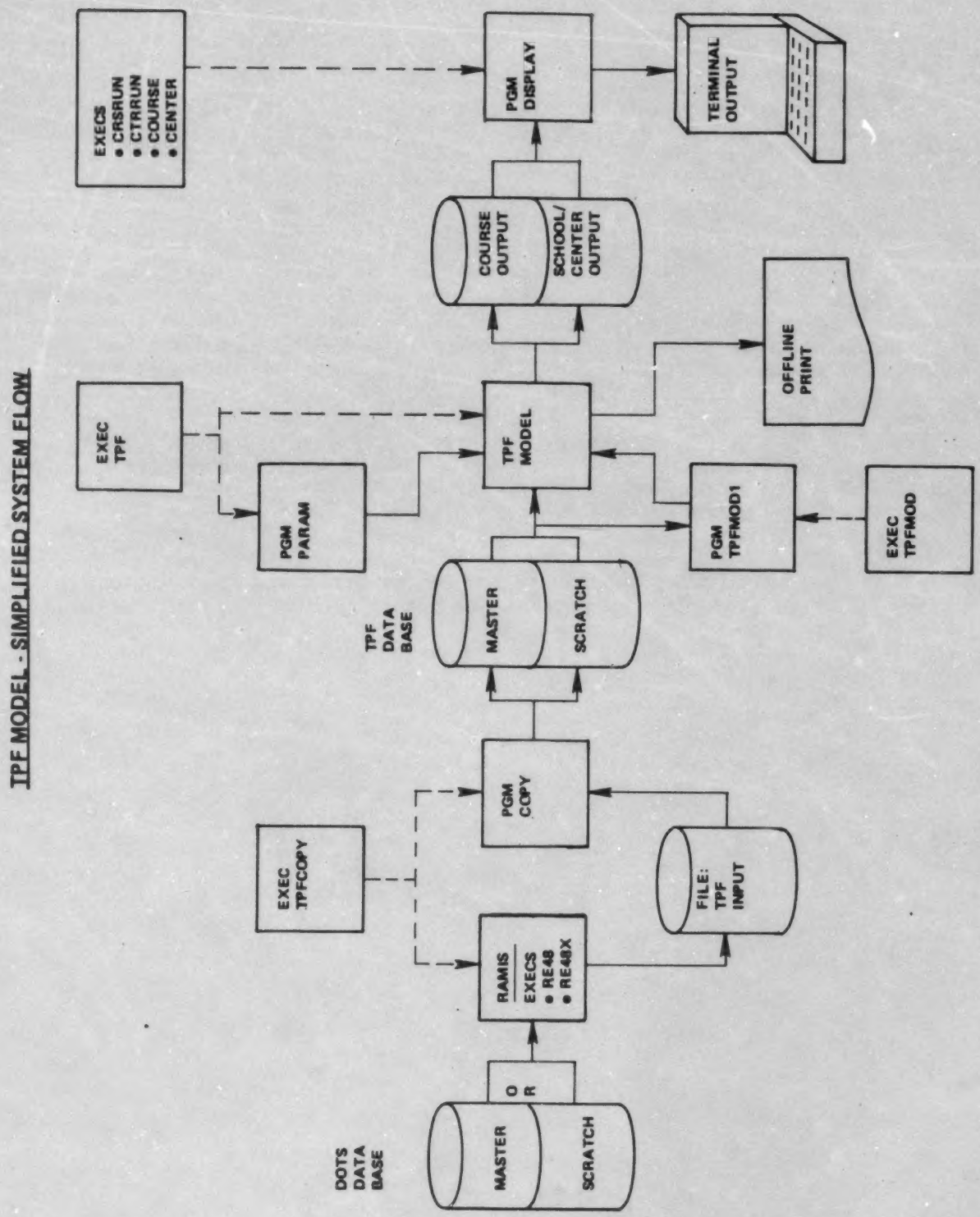
SECTION V

TRAINING PROCESS FLOW MODEL

MODEL DESCRIPTION

The objectives of the Training Process Flow model system are to provide training management with the ability to assess the effects upon the training system of changes in demand, scheduling, and capacities, and to present the model output data so that effective decisions can be made for maximizing the utilization of training command resources. The Training Process Flow model system consists of the TPF model and those support programs and executive which interface with the DOTS data base, provide "what if" capability, control model execution, and provide terminal output capabilities. The overall flow of the TPF model system is shown in Figure V-1. The support programs and their associated executives are described first, with the description of the TPF model last.

SUPPORT PROGRAMS. The TPF support programs fall into three categories: those necessary to interface the TPF model with the DOTS data base, those required to support the "what if" capability of the TPF model, and those required to support selective report capability for terminal operation.



IPF MODEL - SIMPLIFIED SYSTEM FLOW

FIGURE V-1

DATA BASE INTERFACE PROGRAMS

All data used for TPF model execution are stored in the DOTS DATA BASE (master or scratch) using the RAMIS system. A series of executives and programs is provided to facilitate unloading of the RAMIS data base, and recopying these data in a format compatible to the TPF model. The flow of these support programs and executives is shown in Figure V-2. All data used by the DOTS system are stored in the DOTS Master data base. These data can be extracted or modified, and is stored in the DOTS Scratch data base. All procedures concerning the use and manipulation of these data bases are discussed in Section II.

Data from the DOTS data base must be copied into TPF compatible files prior to execution of the TPF model. The executive that controls this function is called TPFCOPY. The logic flow for this executive is shown in Figure V-4. When this executive is called, the parameter MASTER or SCRATCH may be provided to direct execution to the proper data base, if not, it will be asked for. The first step in the executive is to assure that the users core is set to at least 320K Bytes, due to RAMIS requirements. If the core storage provided is not sufficient, an appropriate error message is displayed, core storage is set to the proper value, and the executive terminates. The executive must be restarted, as any set core operation causes a re-IPL of the user's system, which terminates any ongoing operation. The executive tests &INDEX for the number of parameters passed to the executive by the user. If no parameters are provided, the user is prompted to enter MASTER or scratch. The executive then tests for the appropriate response, and will stay in this loop until one of these two keywords is provided.

The executive then attempts to attach the DISK associated with IBM/CKF so that the data base can be accessed. This disk is attached as unit 193 (U) in the writable mode. BRIEF is specified so that no messages are displayed during this attach and &INDEXØ contains a Ø if the attach is successful. If the attach fails, an appropriate message is displayed the the executive terminates, restoring core to 256K.

RAMIS is invoked next, and a FILEDEF command indicates that a disk data set, TPF INPUT, will be used as the temporary storage for the unloaded data base. The executive then runs procedure RE48 RAM if the master data base is to be unloaded, or RE48X RAM if the scratch data base is to be unloaded. These RAMIS procedures are identical with the exception that RE48 RAM accesses file COURSE, or the master file, while RE48 RAM accesses file MODCOURSE, or the scratch file. Two characteristics of these procedures should be mentioned. First, RAMDATA DATA must be specified as "U" to override the default "P" specification, as the DATA BASE files are attached as 193 (U). Secondly, only the file up to the variable UCAP is unloaded for each course.

The format of the data unloaded into file TPF INPUT is shown in Table V-1 with two records per course. Executive TPFCOPY then invokes an executive COPY, which in turn invokes FORTRAN program COPY. This program merely reads the data unloaded from the RAMIS data base now stored as TPF INPUT, and rewrites it as either TPF MASTER or TPF SCRATCH, in a format compatible with the TPF model system. The format for these files is shown in Figure V-3, A and B. The data are stored as two 80 byte records per course. No flow diagrams are shown for either the COPY executive or FORTRAN program due to their simplicity.

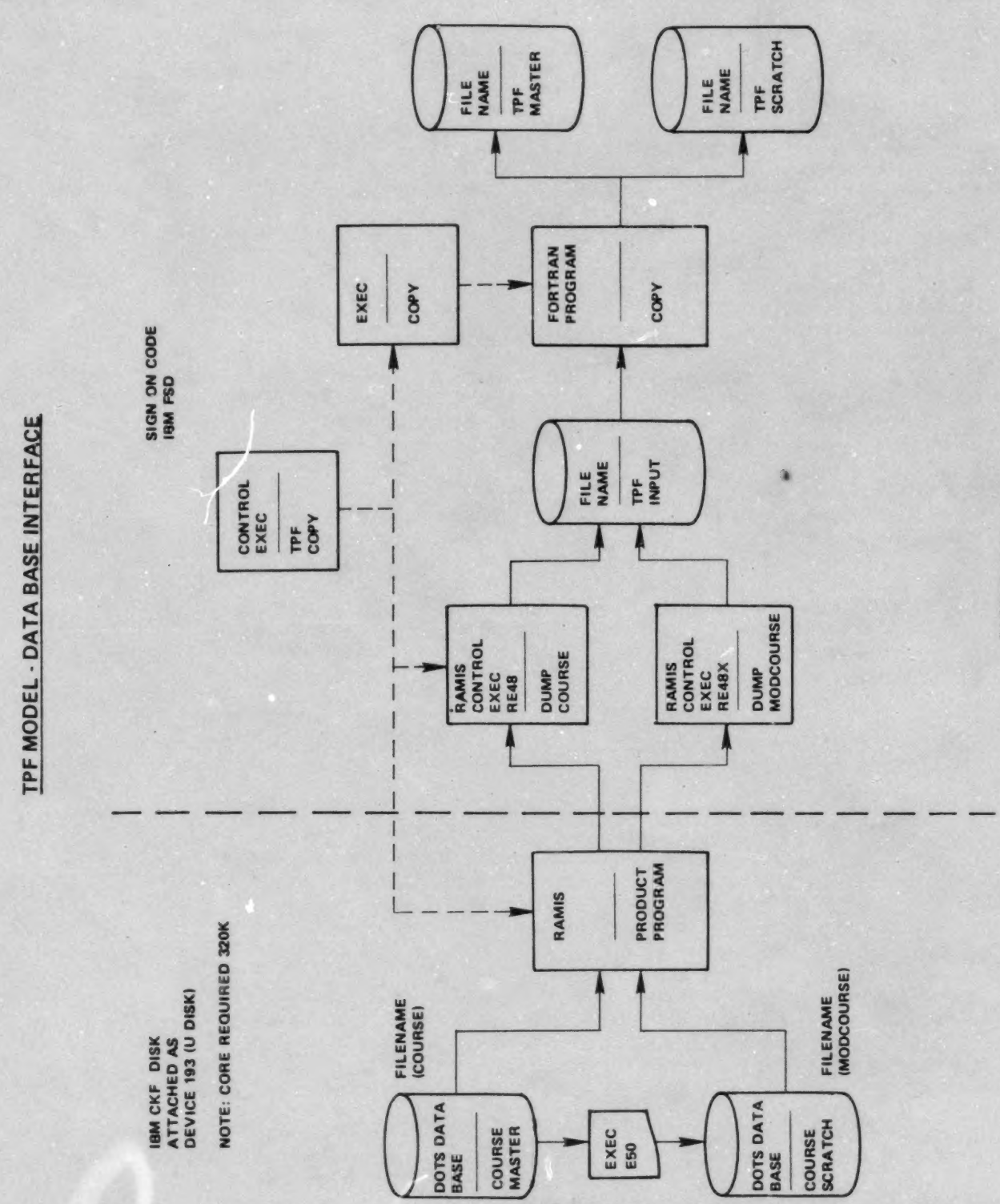

TPF MODEL - DATA BASE INTERFACE
SIGN ON CODE
IBM FSD
IBM CKF DISK
ATTACHED AS
DEVICE 193 (U DISK)
NOTE: CORE REQUIRED 320K
CONTROL
EXEC
TPF
COPY
EXEC
COPY
FILE
NAME
TPF
MASTER
FILE
NAME
TPF
SCRATCH
FORTRAN
PROGRAM
COPY
FILE
NAME
TPF
INPUT
RAMIS
CONTROL
EXEC
RE48
DUMP
COURSE
RAMIS
CONTROL
EXEC
RE48X
DUMP
MODCOURSE
RAMIS
PRODUCT
PROGRAM
FILENAME
(COURSE)
DOTS DATA
BASE
COURSE
MASTER
EXEC
E50
DOTS DATA
BASE
COURSE
SCRATCH
FILENAME
(MODCOURSE)

FIGURE V-2

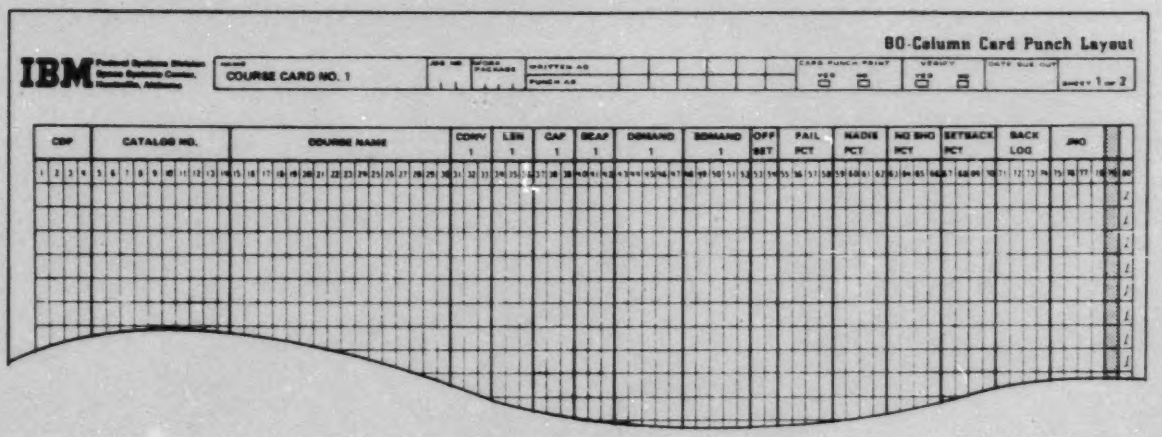
RAMIS DUMP FORMAT (MASTER OR SCRATCH)

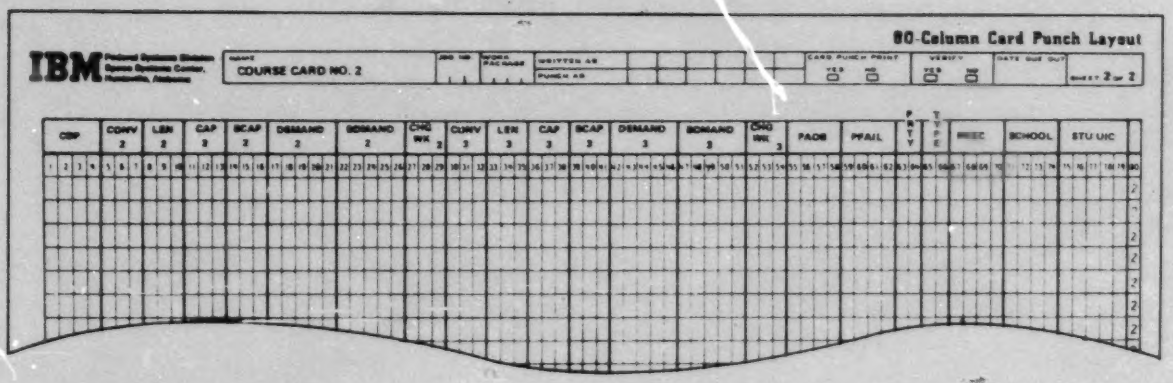
± EBCDIC RECORD LENGTH 8Ø CHARACTERS

FIELD NAME	FORMAT		LENGTH
SCH	A 4		4
CDP	A 4		4
DEL	A 1		1
CTLG	A 1Ø		1Ø
CRS	A 16		16
CONV1	I 3		3
LEN1	F4 .1		4
CAP1	I 3		3
BCAP1	I 3		3
DMD1	I 5		5
BDMD1	I 5		5
OFST	I 2		2
PFAIL	F5 .3		5
NADIS	F5 .3		5
NOSHOW	F5 .3		5
SETB	F5 .3		5
		X	Ø
BKLG	I 4		4
JNO	A 4		4
CONV2	I 3		3
LEN2	F4 .1		4
CAP2	I 3		3
BCAP2	I 3		3
DMD2	I 5		5
BDMD2	I 5		5
CHWK2	I 3		3
CONV3	I 3		3
LEN3	F4 .1		4
CAP3	I 3		3
BCAP3	I 3		3
DMD3	I 5		5
BDMD3	I 5		5
CHWK3	I 3		3
PAOB	I 4		4
AFAIL	F5 .3		5
PRTY	I 2		2
CTYPE	A 2		2
PNEC	A 4		4
UCAP	I 3		3

TABLE V-1

TPF INPUT FORMATS





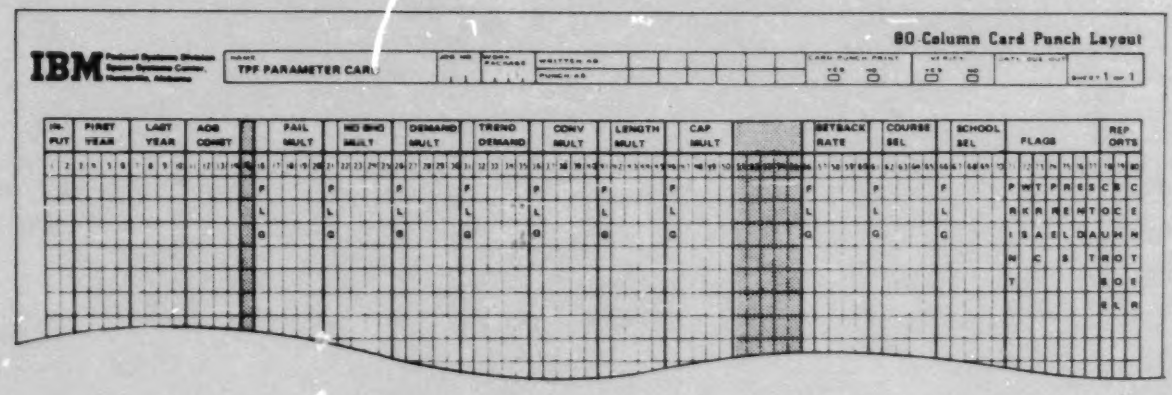


FIGURE V-3

FLOW - EXECUTIVE TPF COPY

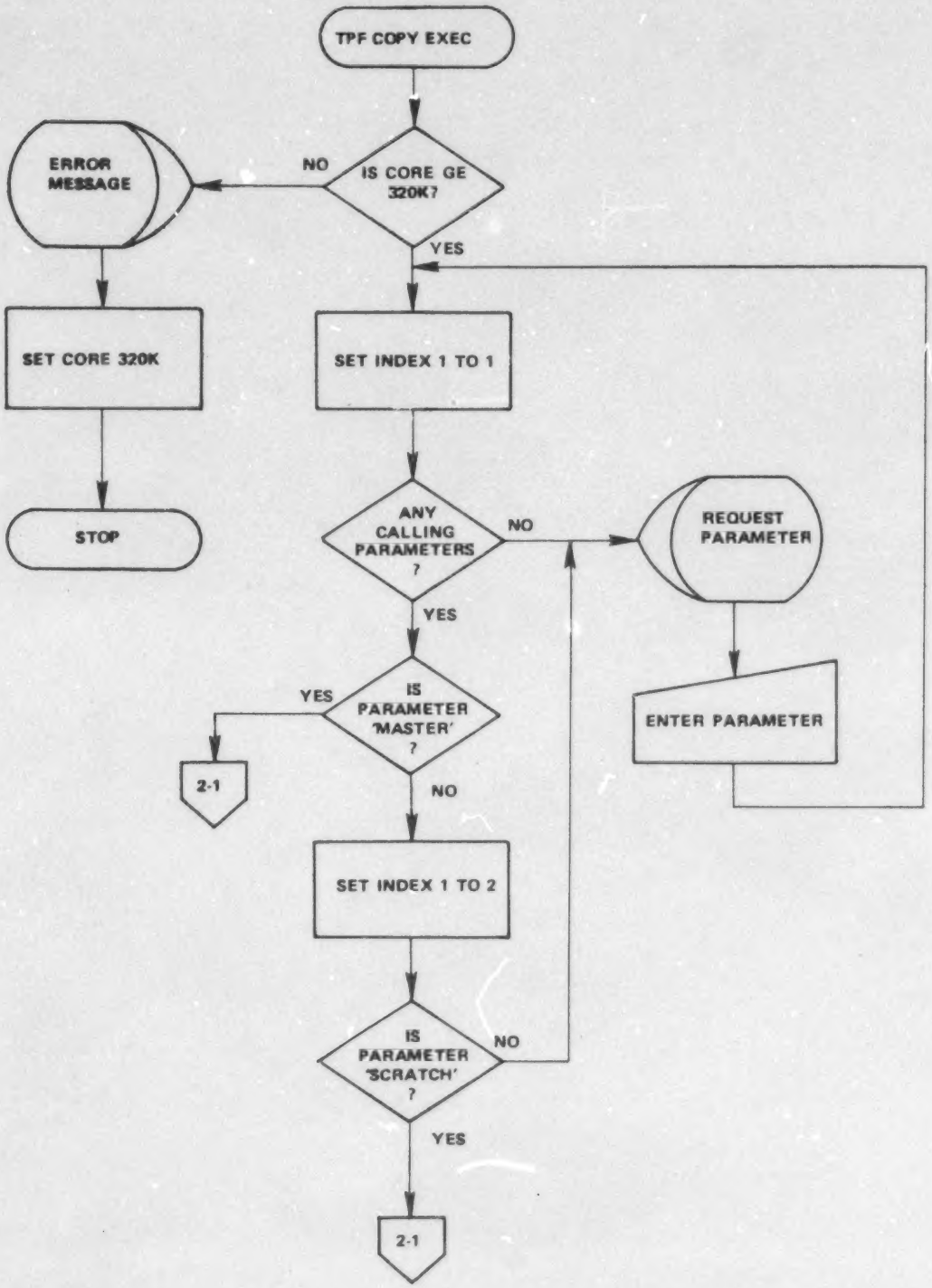


FIGURE V-4

FLOW - EXECUTIVE TPF COPY (CONTINUED)

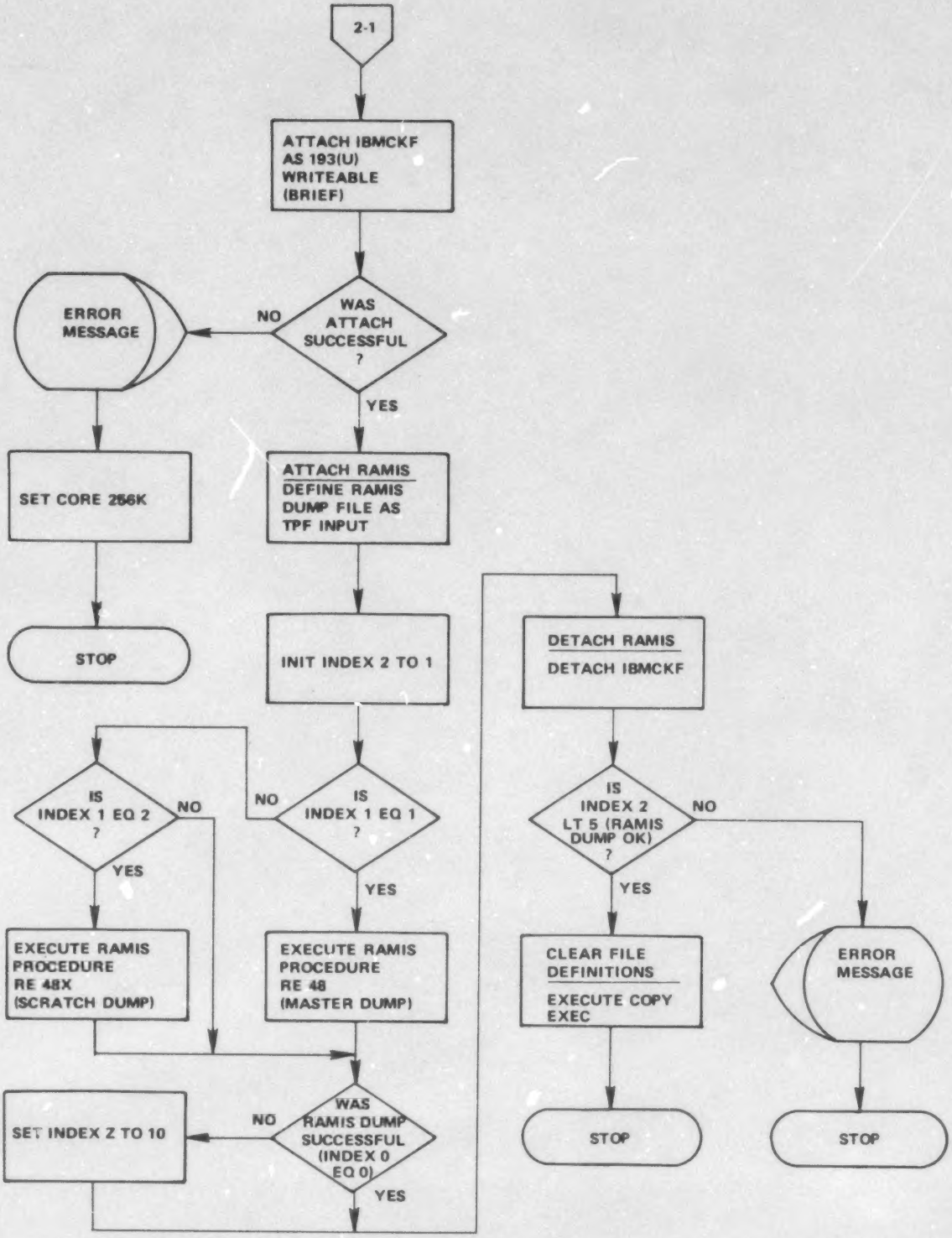


FIGURE V-4 (CONTINUED)

TPFCOPY EXEC

```
&TYPE OFF
&IF &CORE GE 32Ø &GOTO -OK
&PRINT CORE WAS NOT 32Ø K - USER MUST RESTART PROCEDURE
SET CORE 32Ø
&EXIT
-OK &INDEX1 = 1
&IF &INDEX EQ Ø &GOTO -NONE
&IF &1 EQ MASTER &GOTO -RUNIT
&INDEX1 = 2
&IF &1 EQ SCRATCH &GOTO -RUNIT
-NONE &PRINT REPLY MASTER OR SCRATCH
&READ ARGS
&GOTO -OK
-RUNIT ATTACH IBMCKF DOTS 191 AS 193 W BR
&IF &INDEXØ EQ Ø &GOTO -FREE
&PRINT IBMCKF IN USE- WAIT TILL CLEAR AND RETRY
SET CORE 256
&EXIT
-FREE ATTACH RAMIS ALL BR
FILEDEF RAMUTIL1 DSK TPF INPUT
&STACK KT
&INDEX2 = 1
&IF &INDEX1 EQ 1 EXEC RAMISIN RE48 RAM
&IF &INDEX1 EQ 2 EXEC RAMISIN RE48X RAM
&IF &INDEXØ NE Ø &INDEX2 = 1Ø
DETACH RAMIS BR
DETACH 193 BR
&IF &INDEX2 LT 5 &GOTO -RAMOK
&STACK RT
&PRINT FILE NOT DUMPED- RAMIS ERROR
SET CORE 256
&EXIT
-RAMOK FILEDEF * CLEAR
&STACK RT
&IF &INDEX1 EQ 1 EXEC COPY MASTER
&IF &INDEX1 EQ 2 EXEC COPY SCRATCH
&EXIT
```

COPY EXEC

```
&TYPE OFF
FILEDEF 9 DSK TPF INPUT
FILEDEF 1Ø DSK TPF MASTER
&IF &INDEX EQ Ø &GOTO -MASTER
&IF &1 NE SCRATCH &GOTO -MASTER
FILEDEF 1Ø DSK TPF SCRATCH
&PRINT TPF SCRATCH FILE BEING UPDATED
RUN COPY
SET CORE 256
&EXIT
-MASTER &PRINT TPF MASTER FILE BEING UPDATED
RUN COPY
SET CORE 256
&EXIT
```

TABLE V-2

RE48 RAM

```
USE
RAMDATA DATA U
END
DUMP
FILE COURSE
PRINT UCAP AND SAVE
END
```

RE48X RAM

```
USE
RAMDATA DATA U
END
DUMP
FILE MODCOURSE
PRINT UCAP AND SAVE
END
```

COPY FORTRAN

```
C PROGRAM TO CONVERT RAMIS UNLOADED DB TO TPF FORMAT Ø3JUN75
      IMPLICIT INTEGER*2 ($)
      INTEGER*4 $1,$16,$35,$36
      REAL*4 $5,$11,$12,$13,$14,$18,$25,$31,$32
      REAL*8 $2A,$3A,$3B
      I = Ø
    1 READ (9,2,END=999) $36,$1,$2A,$2B,$3A,$3B,$4,$5,$6,$7,$8,$9,$1Ø
     *,$11,$12,$13,$14,$15,$16,$17,$18,$19,$2Ø,$21,$22,$23,$24,$25,$26
     *,$27,$28,$29,$3Ø,$31,$32,$33,$34,$35
    2 FORMAT (2A4,1X,A8,A2,2A8,I3,F4.1,2I3,2I5,I2,4F5.3,/
     * I4,A4,2(I3,F4.1,2I3,2I5,I3),F4.1,F5.3,2A2,A4,4X)
      WRITE (1Ø,516) $1,$2A,$2B,$3A,$3B,$4,$5,$6,$7,$8,$9,$1Ø,
     * $11,$12,$13,$14,$15,$16,$1,$17,$18,$19,$2Ø,$21,$22,
     * $23, $24,$25,$26,$27,$28,$29,$3Ø,$31,$32,$33,$34,$35,$36
  516 FORMAT (A4,A8,A2,2A8,I3,F3.Ø,2I3,2I5,I2,4F4.3,I4,A4,2X,/,A4,
     * 2(I3,F3.Ø,2I3,2I5,I3),2F4.1,2A2,2A4)
      I = I + 1
      GO TO 1
  999 WRITE (6,99) I
   99 FORMAT (/' END TPF DB FILE UPDATE-',I3,' COURSES UPDATED'/)
      STOP
      END
```

TABLE V-3

"WHAT IF" SUPPORT PROGRAMS

The TPF model user normally will use the TPF system first to examine a school or center under normal, or baseline, conditions, and then to modify certain input parameters and observe the changes to the outputs that occur. Three approaches are available to the user to modify these input data. See Figure V-5. The first is to use the RAMIS facilities to modify the baseline data and create a DOTS scratch data base. The use of RAMIS to accomplish this is discussed in Section II of this report. The second method is to execute program TPFMOD1, which allows the user to make modifications to individual courses for study, or to insert now or upcoming courses in order to observe their effects on the overall system. Finally, program PARAM can be used to make modifications to a certain type of data for all courses; i.e., reducing the demand for all courses by 10%.

COURSE MODIFICATIONS. Executive TPFMOD, in conjunction with program TPFMOD1, is used to control the "what if" capability of the TPF system for modifications to individual courses or the study or inclusion of new courses to the center.

Executive TPFMOD. Executive TPFMOD is quite straight forward, and its flow is shown in Figure V-6. This executive sets up the program to access either the TPF MASTER or TPF SCRATCH data base created by copying and reformatting the data stored in the DOTS data base. A third file, TPF VFY, shown in Figure V-5. is available to the TPF model for verification purposes, but cannot be accessed by TPFMOD1. Three features of executive TPFMOD should be explained. First, &INDEX contains the number of parameters passed the executive during the call (or parameter read), and &1 ... &10 contain the parameters. It is important to test that there are parameters before testing the value of the parameters as the executive will ABEND on a value test if none are present. Secondly, &INDEX0 contains the condition code of the last program run. In this case, it is used to test the condition code of the STOP command in program TPFMOD1, which will be 0 if the user desires to save the data created, and 999 if it is not to be saved. Finally, program TPFMOD1 overrides the normal error checking for input/output errors of the System/370, and this error checking must be reset by the SET SPIE ON command to allow normal processing to continue. Failure to reset this SPIE may cause subsequent programs to loop during execution of I/O commands, resulting in excessive consumption of computer time and charges.

Program TPFMOD1. Program TPFMOD1 is actually a load module obtained by combining main program TPFMOD with subroutines COMND, CONVRT, MSG, DECODE, and SHOW. The FORTRAN program TPFMOD is discussed first, with each subroutine discussed separately.

Program TPFMOD has two purposes. The first is to allow the user to access specific courses from the TPF data base (master or scratch), make alterations to any of the data elements, and to store the modified course data for TPF model execution. The second purpose is to allow the user to create new courses, prompted by the program, and again store them for TPF execution. Program TPFMOD (main) controls the sequence of execution to accomplish this. This control program can be broken into three parts: The first part controls the "COPY" option, the second controls the "NEW" option, while the third section controls the "MOD" option. Several features are consistent throughout this main program and the various subroutines. First, all data set input uses the direct access method of FORTRAN read. This is to allow the user to skip through the files without

TPF MODEL - "WHAT IF", AND OUTPUT DISPLAY

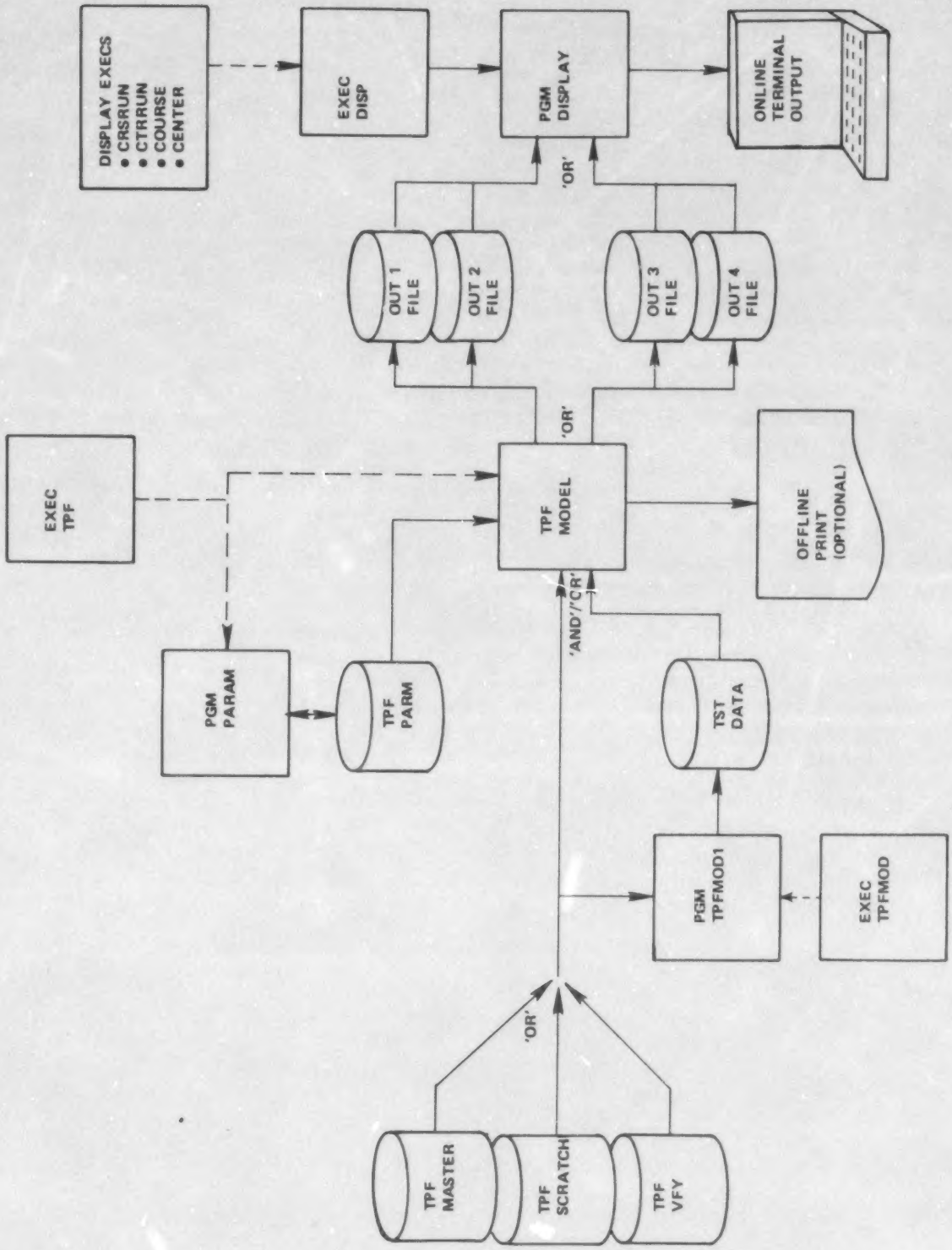


FIGURE V-5

the need to open and close them on each read. A CALL ERRSET command is declared at the beginning of the program to override error checking during this I/O operation. Should the user specify a course CDP number not in file, inline subroutine RANGE will be invoked to recover the error and allow processing to continue.

Secondly, most terminal I/O uses the 10A1 format. This is to allow the user to intermix alpha, numeric, and decimal characters on the same read operation and to allow entry of a variable number of parameters during the same read. Subroutine CONVRT is used to change the results of this read back to internally usable data. Finally, during any terminal read operation, terms such as HELP, QUIT, SHOW, DONE, etc., may be entered, and subroutine MSG will provide the proper action.

The COPY option allows the user to copy all data for a particular course into core storage for subsequent modification and finally execution by the TPF model. The course is called by the CDP number, and either file TPF MASTER or TPF SCRATCH is searched for that course. The course data are stored in the order they are accessed rather than by CDP number. This is to allow evaluation of a course under several conditions during a single model run, hence the same course can be copied several times. However, due to the pointer method of storage, the sequence of operation must be COPY, MODIFY, COPY, MODIFY...as it is impossible to modify course data after a new course has been read in.

The NEW option allows the user to create new courses in the file. The user is prompted for the proper data and format for only those data elements required for TPF execution. This prompting is accomplished through a DO loop, which repetitively calls subroutine DECODE, handing off the proper parameter. The course counter is incremented once a new course has been inserted in the file.

The MOD option allows the user to modify data for a course inserted in the file, whether through the COPY or NEW option. The user is prompted to enter a field name. Subroutine DECODE then validates the field name and prompts for the data in the proper format. Subroutine DECODE also accepts the field name, followed by a blank and then the data to speed up the operation by an experienced operator. The field names in this program are identical to the field names used in the DOTS data base, and may be displayed by entering HELP and asking for the field names.

The wrapup of this program allows the user to bypass storage of the file created if errors occurred, and also supports the HELP option. The HELP option allows the user to display the field names or commands available, without reference to an operation manual. Processing returns to the point at which HELP was entered upon completion of this option.

Subroutine DECODE has three purposes. The first is to support the NEW option by prompting for proper field data. The second is to support the MOD option, by verifying field names, and prompting for and verifying the field data. Finally, the subroutine supports the STAT command, and will display the current value of any field for the course in process. Communication with the main program, or other calling subroutines, is accomplished through parameters passed with the subroutine call for control and return codes, and through COMMON storage for data elements. DECODE is called in the form:

CALL DECODE (I, N, RESP, IN, IN2)

where: All data elements are of the form INTEGER*2

I contains the valid field number if known (usually through the new option), a zero if it is not known, or a 9999 if the STAT option is in effect (from subroutine SHOW).

N is a pointer at the data in storage for the course being processed.

RESP contains the return code (0 for valid operation, 99 for an invalid operation, or the subroutine MSG return code if the operator keyed in a valid message during execution.

IN and IN2 are 5 element arrays containing the data to be converted. IN contains the field name, while IN2 contains the data.

Subroutine DECODE contains the type of data (integer, alpha, or decimal) for each field in an array called TYPE, and the limiting value for each field in an array called SIZE. This allows error checking of data elements and allows prompting of the user if no data are entered with the field name or an error occurs. During execution of subroutine DECODE, if any data entered are invalid or not recognized, subroutine MSG is called to ascertain if a valid message was entered, and if so, subroutine DECODE terminates, using the message number as a return code.

Subroutine CONVRT is a FORTRAN subroutine called by TPFMOD MAIN and subroutine DECODE, as well as other programs elsewhere in the TPF system. The purpose of CONVRT is to convert data read into storage in the A format to a data form compatible with internal processing. The A format is used in the TPF system for terminal inputs to allow intermixing of data, messages, and field names, allow blank delimited input, to allow intermixing of integer and decimal data, and to eliminate the need for right or left justification for data input. This subroutine is called in the form:

CALL CONVRT (IN, ANS, CODE, DECMAL)

where: IN is in the form INTEGER*2 and is a 5 element array containing the data to be converted.

ANS is in the form INTEGER*4 and contains the integer conversion.

CODE is in the form INTEGER*2 and contains the return code for the subroutine (0 for integer conversion, 5 for decimal conversion, 10 for invalid conversion).

DECMAL is in the form REAL*4 and contains both the real and fractional portion of the answer for a decimal conversion.

Basic operation of this subroutine is quite simple. Each digit of the number to be converted was read into storage using the A1 format. Thus, the number 1 is stored in an INTEGER*2 location as hexidecimal F140, rather than the desired 0001, and is thus not valid for processing. To convert each digit to a processable number the value F000 is first subtracted from the number, in this case

leaving 0140. The number is then divided by 256, which causes a shift right 2, leaving 0001. An equivalence statement is used, making the work area for this operation an INTEGER*4, to prevent underflows. Traps are provided to recognize multiple decimals. The five individual values decoded are then combined into a single decimal or integer value.

Subroutine MSG is a short FORTRAN subroutine used to decode one of 9 valid text messages passed from the calling program or subroutine. It is called in the form:

CALL MSG (IN, CODE)

where: IN is in the form INTEGER*2 and is a 5 element array containing the message to be decoded.

CODE is in the form INTEGER*2 and contains one of the following return codes:

QUIT(1), DONE(2), HELP(3), STAT(4), SHOW(5),
COPY(6), NEW(7), ALL(8), MOD(9), NOT RECOGNIZED(99).

Subroutine SHOW is a FORTRAN subroutine to provide display of all elements for the course in process. This subroutine is called in the form:

CALL SHOW (N)

where: N is in the form INTEGER*2 and contains a pointer to the course in the data matrix in process.

The user is allowed to specify a specific field to be displayed, or call for all fields by means of a CR when prompted. The format of the data displayed is similar to that used for the TPF model. However, the student backlog figure used in the TPF model is found during model initialization, while the figure used here is calculated, with the possibility that in a few cases the backlog figure may differ slightly. No return code is provided for in this subroutine, hence messages cannot be decoded. This subroutine also has the BLOCK DATA subroutine attached, which accomplishes initialization of all data in COMMON storage prior to program execution.

Subroutine COMND is an assembly language subroutine whose execution causes the screen of the Hazeltine display to be cleared. It has no effect on most other terminals. It is called in the form:

CALL COMND (I, J)

where: I and J are INTEGER*2 variables, I must contain a 6 to cause a clear operations and J contains the return code (0 for valid operation, 1 or 2 for error).

The return code is not tested. This subroutine is used by most FORTRAN and PL/I programs in the TPF system.

ACROSS THE BOARD MODIFICATIONS. Program PARAM, in conjunction with executive

TPF, is used to support "what if" capability of the TPF system for "across the board" modifications to input data. This is accomplished through manipulation of data set TPF PARM which is an input to the TPF model, and through the use of FILEDEF commands in the executive to direct the input/output for the TPF model.

Executive TPF. Executive TPF controls execution of both program PARAM and the TPF model. The executive requests which data sets are to be used as input and sets up the necessary flags in &INDEX1 and &INDEX2. An option exists to enter the word SAVE as the second parameter when the input file is requested, and if present, will set a flag in &INDEX3. Normally center and school output data for the TPF model will be written in OUT3 FILE, and detailed course data will be written in OUT4 FILE, overwriting data already written there. If the SAVE option is specified, output will be directed to OUT1 FILE for center and school data, and OUT2 FILE for course data. Also, the TST DATA file, created by program TPFMOD1 can be concatenated with any other input data set at this time.

Program PARAM. Program PARAM is a PL/I program called by the executive TPF, and is used to modify the contents of data set TPF PARM. This data set contains a single 80 byte record that is used as input data by the TPF model to control execution. The format of this data set is shown in Figure V-3 C. This method of control for the TPF model was chosen to facilitate off-line or batch execution, as well as on-line terminal applications. The program reads the current record from data set TPF PARM, displays the active elements, allows the user to make changes, and then rewrites the record. The first four elements (input, first year, last year, and AOB constant) are always active and their values will be displayed. The next 10 fields are active only if a number greater than zero appears in the flag column. Finally, the last 10 fields contain flags, and an indication will be displayed if their value is non-zero. Use of this parameter data are discussed in the TPF LOGIC DESIGN section. The program runs in a manner similar to the other support programs in this section, and accepts most commands, such as HELP,QUIT, and SHOW as the other program. Internally, most of the field location and data assignment tasks are done using the substring feature of the PL/1 language. This is to allow the variable display format desired for the active fields, and assist in decoding and locating fields for data entry. The COMND subroutine is used to provide the clear display feature at the beginning of the program and PL/1 subroutine IHESARC is used to pass a return code to bypass execution of the TPF model if the user so directs.

FLOW - EXECUTIVE TPFMOD

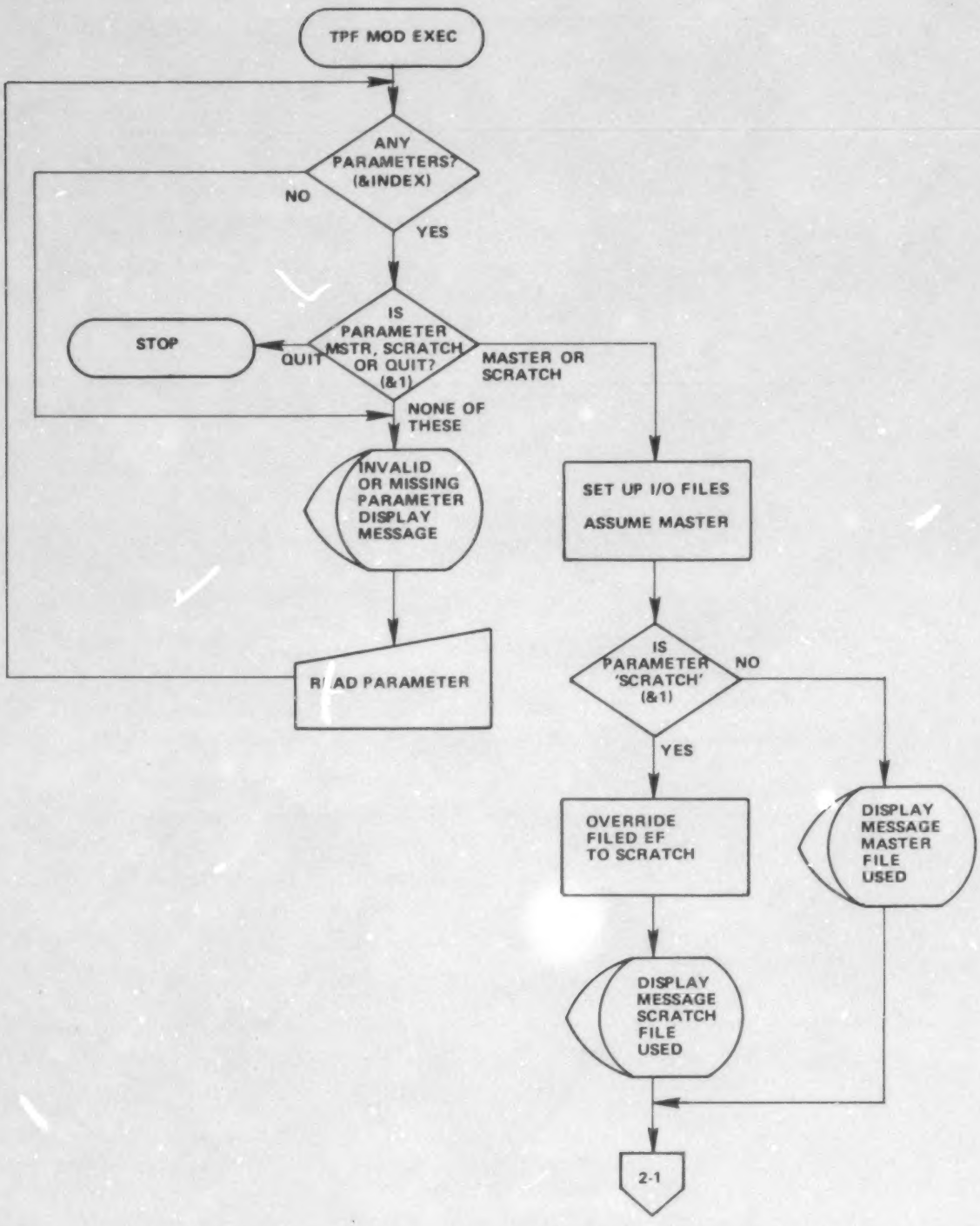


FIGURE V-6

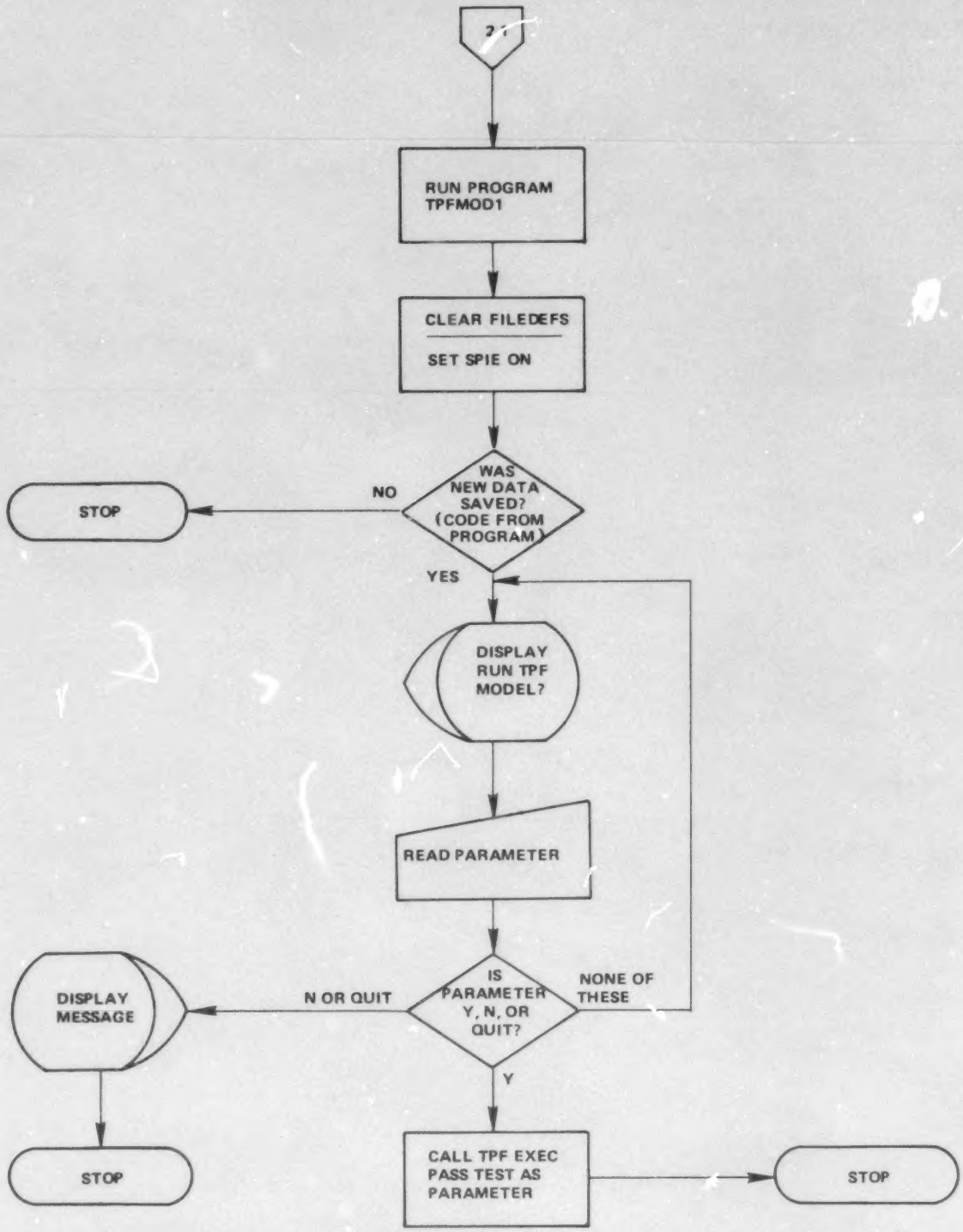


FIGURE V-6 (CONTINUED)

CHART TITLE - PROCEDURES

AUTOFLOW - PROGRAM TPFMOD

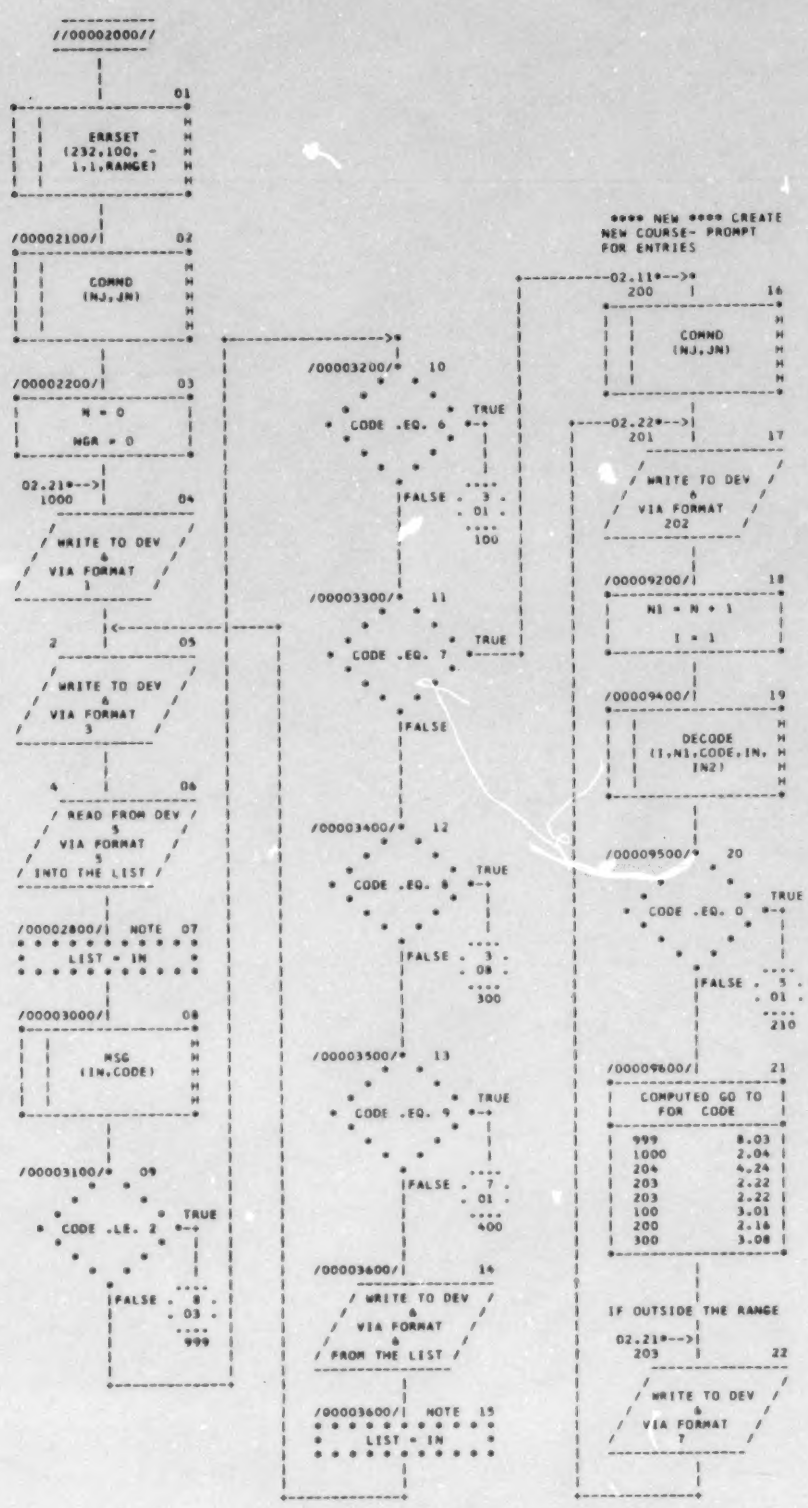


FIGURE V-7

465

CHART TITLE - PROCEDURES

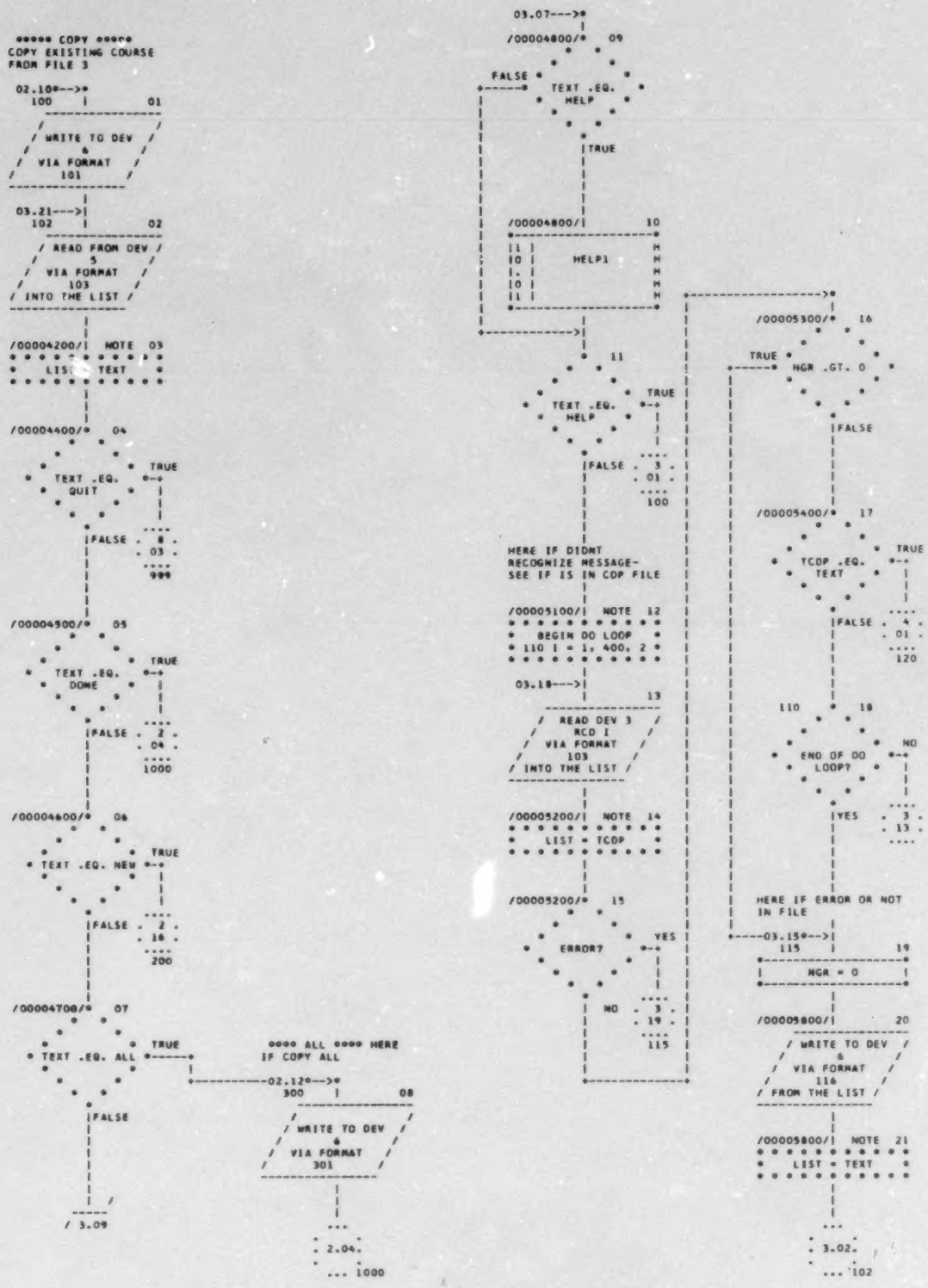


FIGURE V-7 (CONTINUED)

CHART TITLE - PROCEDURES

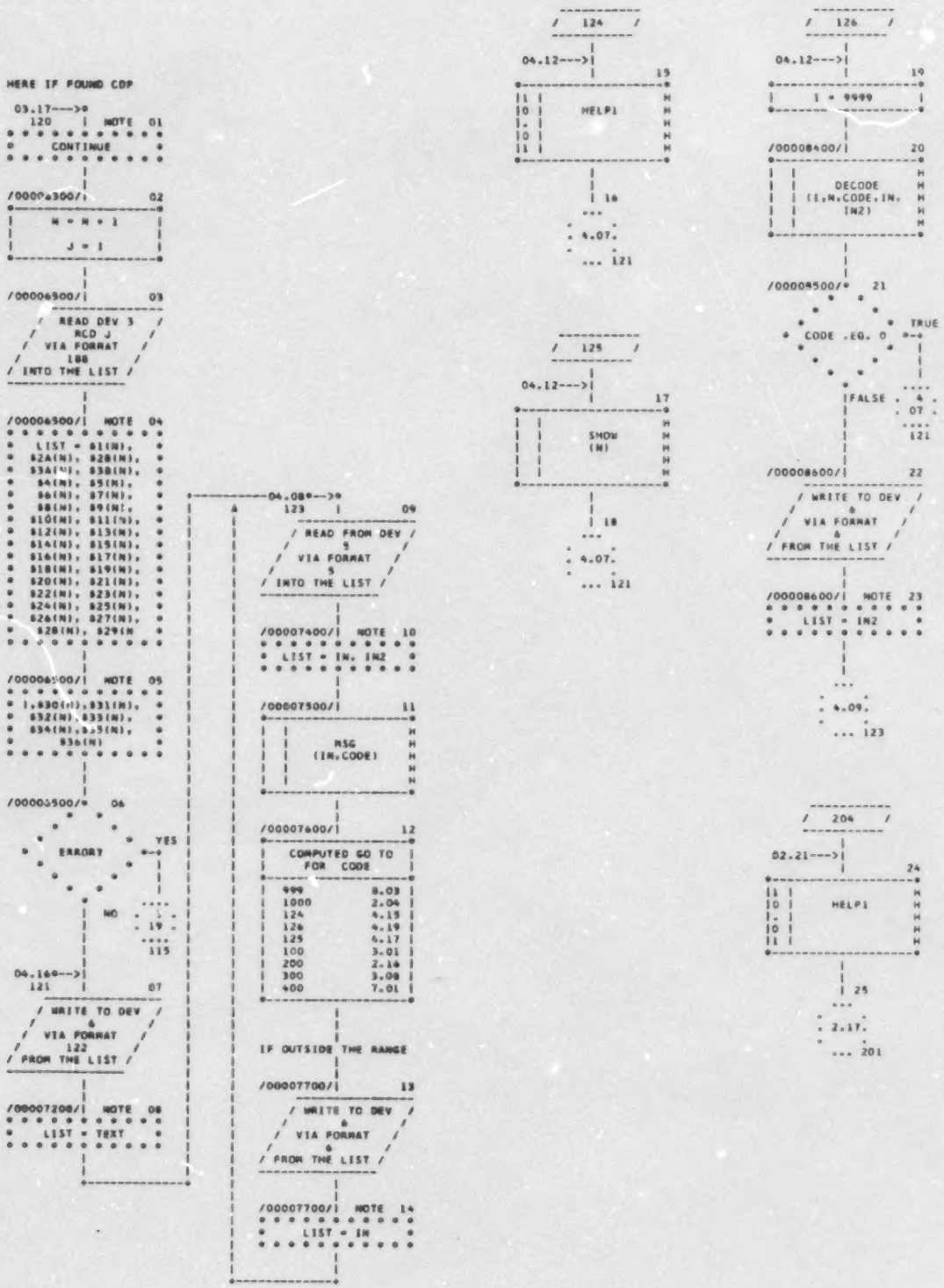


FIGURE V-7 (CONTINUED)

CHART TITLE - PROCEDURES

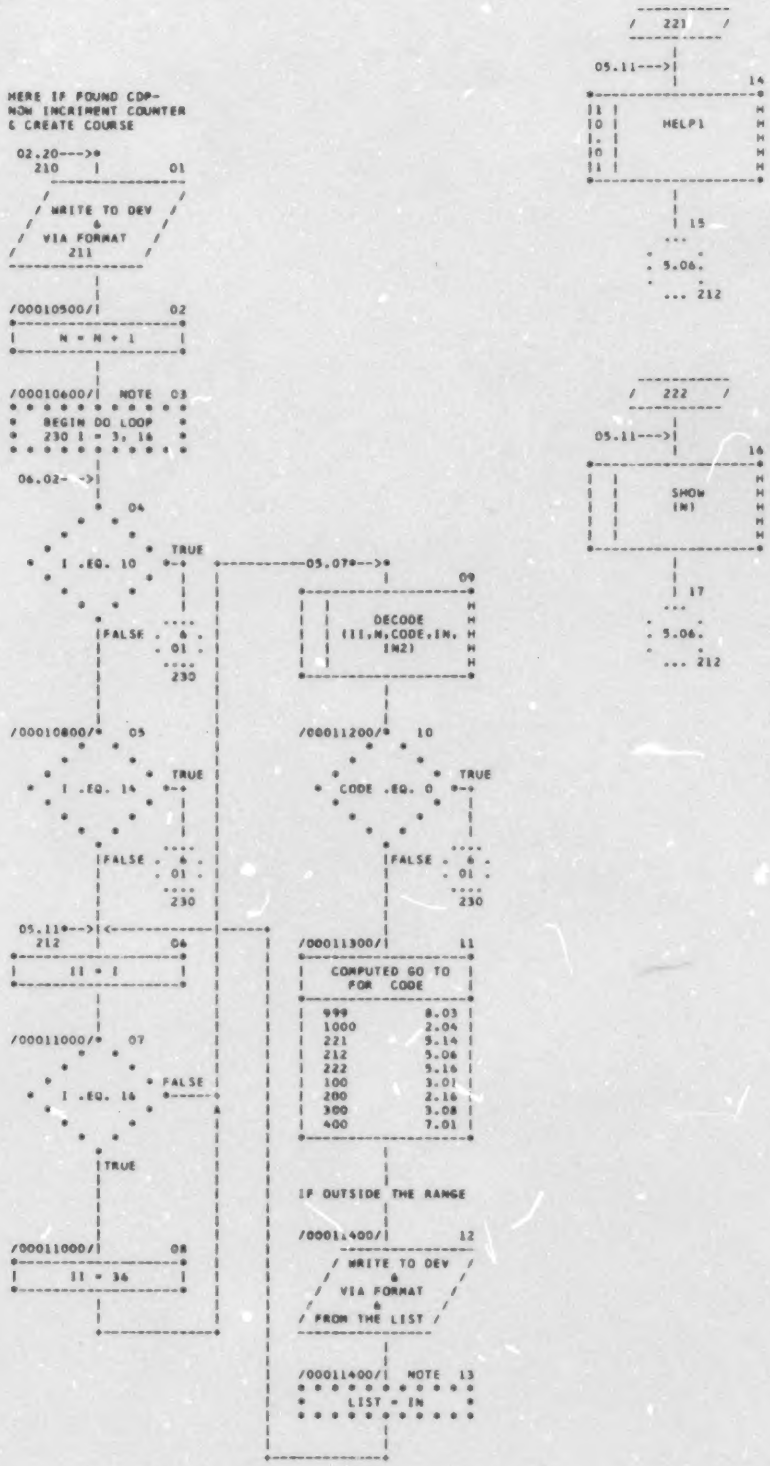


FIGURE V-7 (CONTINUED)

CHART TITLE - PROCEDURES

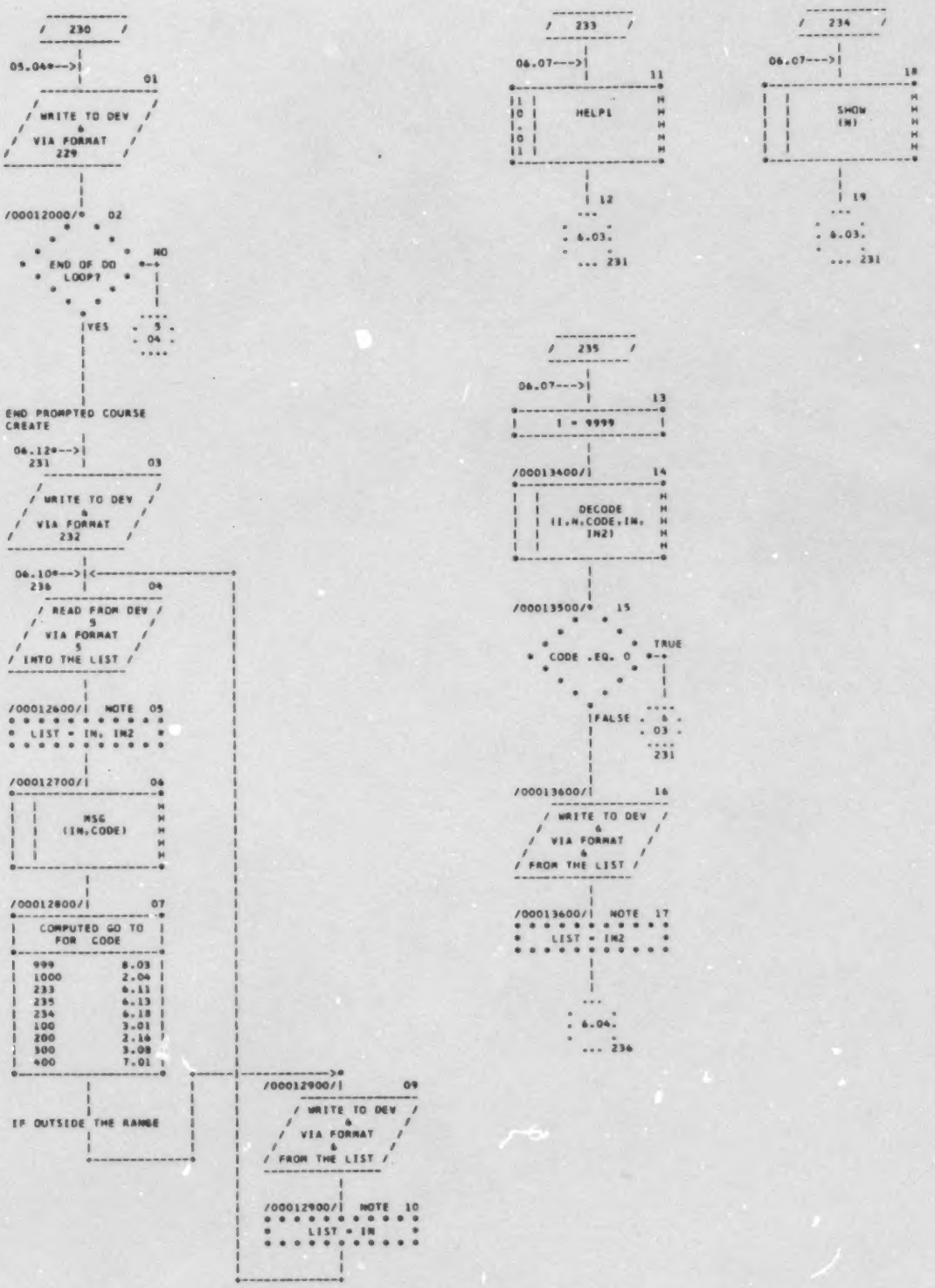


FIGURE V-7 (CONTINUED)

CHART TITLE - PROCEDURES

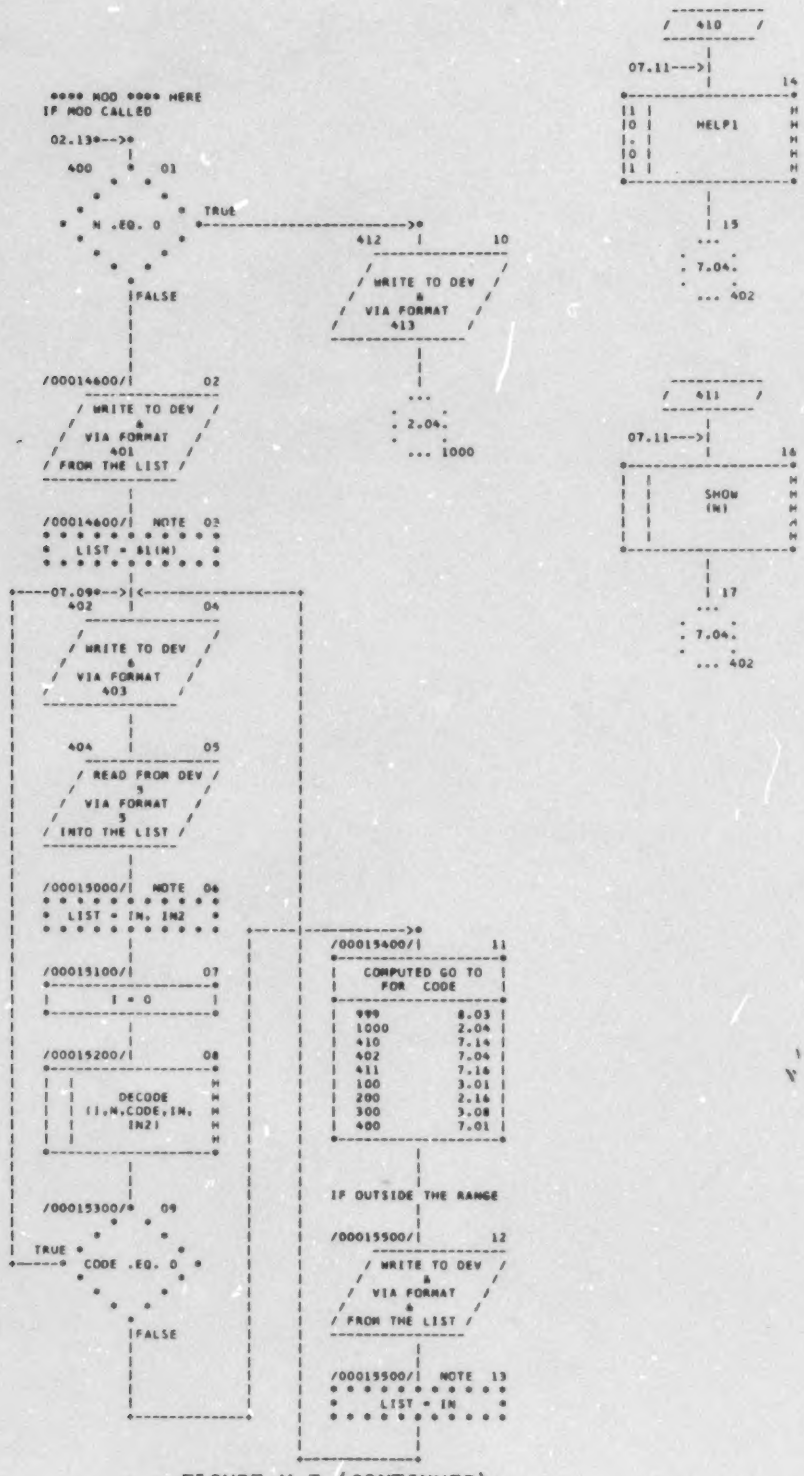


FIGURE V-7 (CONTINUED)

CHART TITLE - PROCEDURES

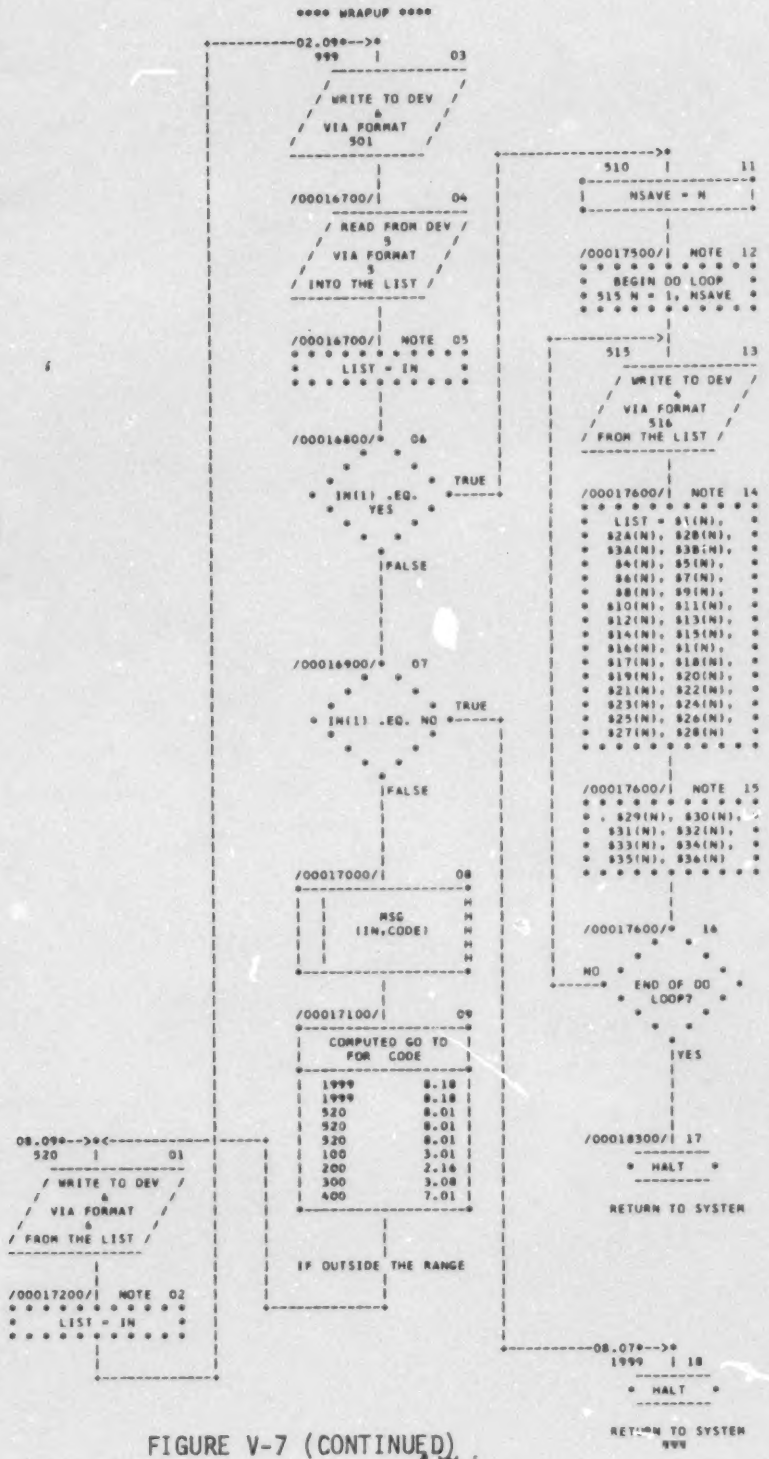


FIGURE V-7 (CONTINUED)

471

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      IMPLICIT INTEGER*2 (8)
      INTEGER*4 81,816,835,836
      REAL*4 85,811,812,813,814,818,825,831,832
      REAL*8 82A,83A,83B
      INTEGER*2 YES/'Y'/, NO/'N'/
      COMMON /A/ 81(144),82A(144),83A(144),84(144),85(144),86(144)
      COMMON /A/ 87(144),88(144),89(144),810(144),811(144),812(144)
      COMMON /A/ 813(144),814(144),815(144),816(144),817(144),818(144)
      COMMON /A/ 819(144),820(144),821(144),822(144),823(144),824(144)
      COMMON /A/ 825(144),826(144),827(144),828(144),829(144),830(144)
      COMMON /A/ 831(144),832(144),833(144),834(144),835(144),836(144)
      COMMON /A/ 828 (144),838(144)
      INTEGER*2 IN(5) /5*0/, IN2 (5) /5*0/, CODE, NJ/6/,JN
      COMMON NGR
      INTEGER*4 NEW /'NEW '/, QUIT /'QUIT'/, DONE /'DONE'/, TEXT, TCDP
      DEFINE FILE 3 (500,80,E,J3)
      EXTERNAL RANGE
1     FORMAT (/' TPF COURSE CREATE/MODIFY PROGRAM'/)
3     FORMAT (' +ENTER ''NEW'' OR ''COPY'' '/)
5     FORMAT (10A1)
6     FORMAT (1X, 5A1, ' +INVALID- RETRY'/)
101   FORMAT (/ ' +COPY EXISTING COURSE- ENTER CDP')
103   FORMAT (A4)
116   FORMAT (1X, A4, ' +NOT IN FILE- RETRY OR ENTER DONE'/)
188   FORMAT (A4,A8,A2,2A8,I3,F3.1,2I3,2I5,I2,4F4.3,I4,A4,2X,/,4X,
       2(I3,F3.1,2I3,2I5,I3),2F4.1,2A2,2A4)
122   FORMAT (/ ' CDP ', A4, ' +COPIED- ENTER ''DONE'' OR ''MOD'' ')
202   FORMAT (/' CREATE NEW COURSE'/)
7     FORMAT (' +INVALID- RETRY'/)
211   FORMAT (/' ENTER FOLLOWING DATA TO CREATE COURSE'/)
229   FORMAT (' ')
232   FORMAT (/' KEY DATA ENTERED FOR CREATE- ENTER ''MOD'' OR ''DONE''
      ')
301   FORMAT (/' ''ALL'' OPTION NOT SUPPORTED THIS PROGRAM' /)
401   FORMAT (/' MODIFY DATA FOR CDP ',A4)
403   FORMAT (/' +ENTER FIELD TO BE CHANGED')
413   FORMAT (' MOD INVALID- NO COURSE IN FILE'/)
501   FORMAT (/' SAVE NEW DATA BASE? (Y OR N)')
516   FORMAT (A4,A8,A2,2A8,I3,F3.0,2I3,2I5,I2,4F4.3,I4,A4,2X,/,A4,
       2(I3,F3.0,2I3,2I5,I3),2F4.1,2A2,2A4)
```

FIGURE V-7 (CONTINUED)

CHART TITLE - SUBROUTINE HELP1

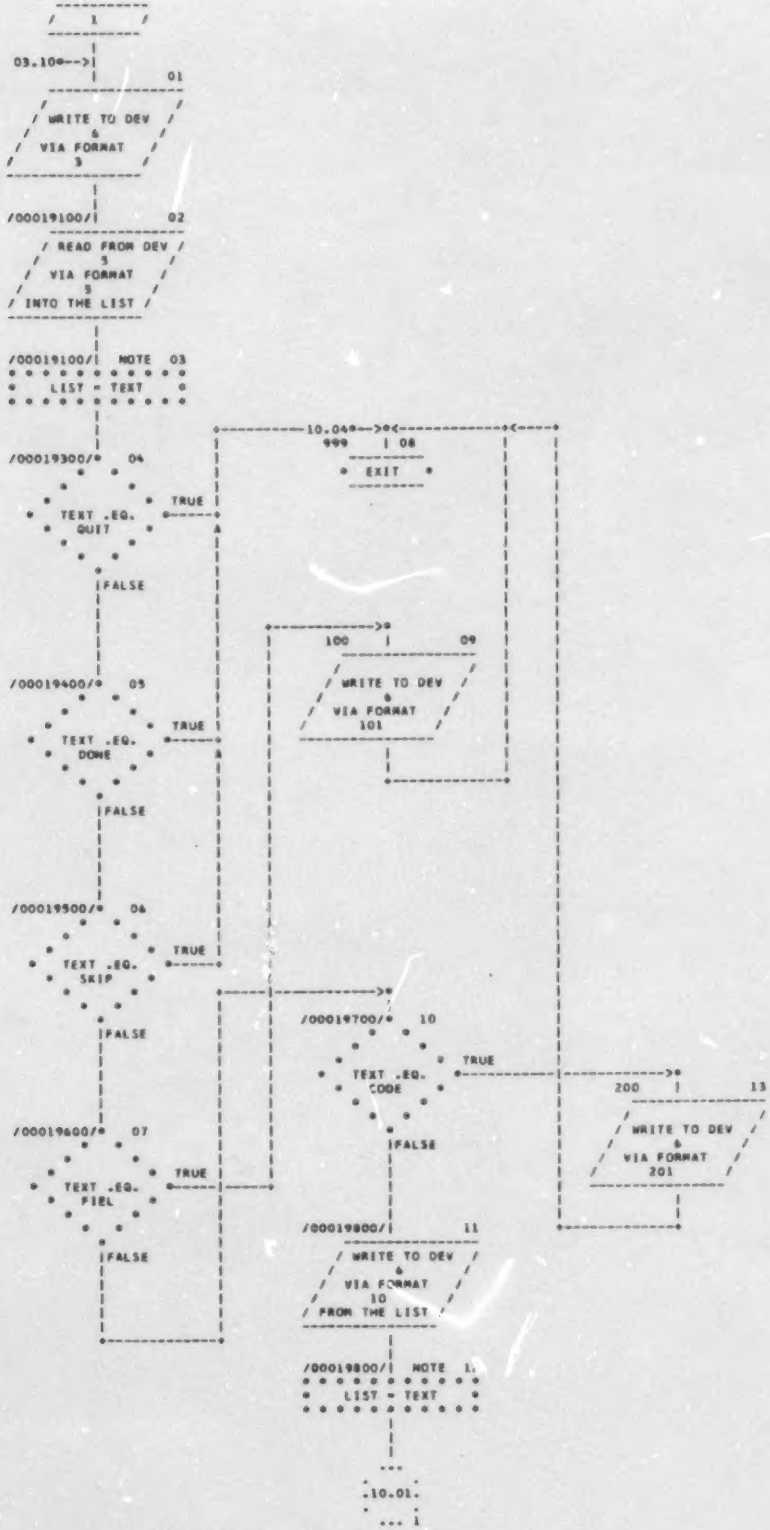


FIGURE V-7 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      INTEGER*4 TEXT,QUIT/'QUIT'/,SKIP/'SKIP'/,FIEL/'FIEL'/,CODE/'CODE'/
      INTEGER*4 DONE /'DONE'/
3     FORMAT (/' +HELP CALLED- ENTER FIELD, CODES, OR SKIP')
5     FORMAT (A4)
10    FORMAT (1X, A4, ' INVALID- RETRY')
101   FORMAT (///' TPF FIELDS ARE:'/
      ' CDP    CONV1  LEN1   CAP1   BCAP1  DMD1   BDMD1  OFST'/
      ' CTLG   CONV2  LEN2   CAP2   BCAP2  DMD2   BDMD2  CHWK1'/
      ' CRS    CONV3  LEN3   CAP3   BCAP3  DMD3   BDMD3  CHWK2'/
      ' SCH    PFAIL  NADIS  NOSHO  SETB   BKLG'/
      ' JND    PNEC   PAJB   AFAIL  PRTY   CTYPE'//)
201   FORMAT (//' VALID CODES ARE:'/
      ' QUIT  DONE  HELP  STAT  SHOW  COPY  NEW  ALL  MOD'//)
```


CHART TITLE - SUBROUTINE RANGE(KK,LL)

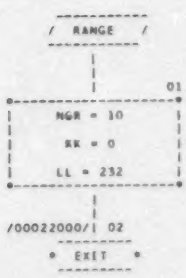


FIGURE V-7 (CONTINUED)

CHART TITLE - SUBROUTINE DECODE(I,N,RESP,IN,IN2)

AUTOFLOW - SUBROUTINE DECODE

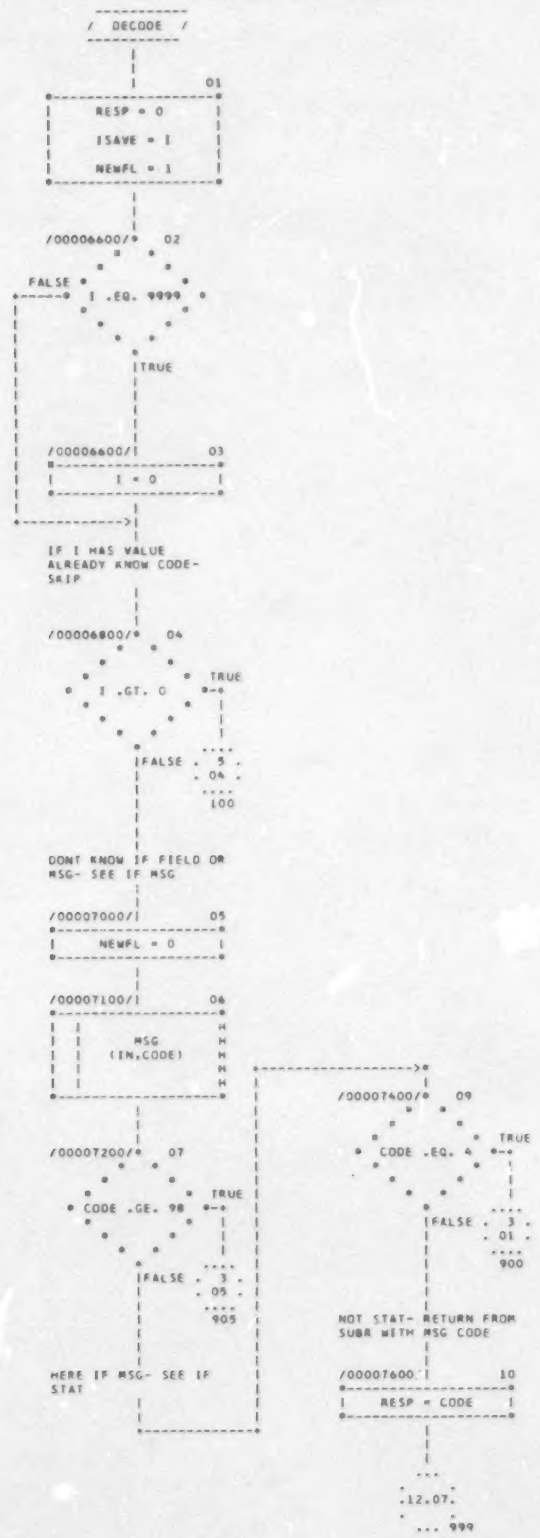


FIGURE V-8

CHART TITLE - SUBROUTINE DECODE(I,N,RESP,IN,IN2)

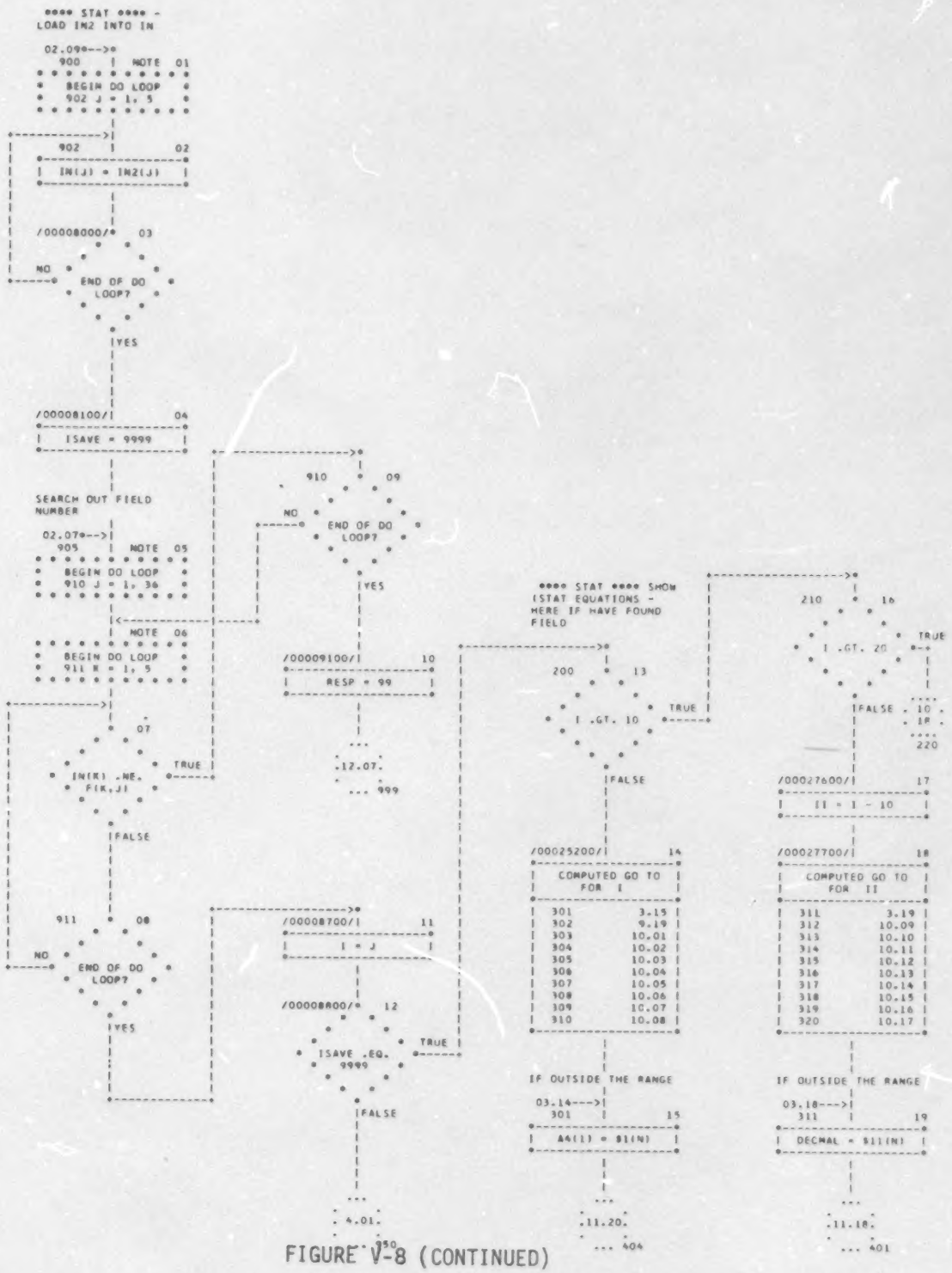


FIGURE V-8 (CONTINUED)

CHART TITLE - SUBROUTINE DECODE(I,N,RESP,IN,IN2)

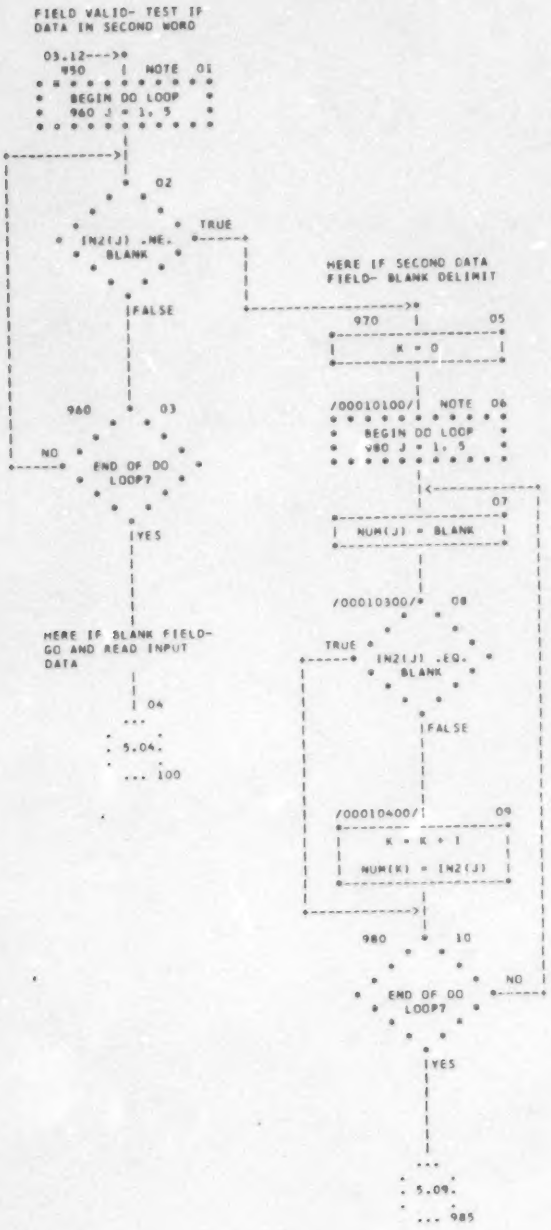


FIGURE V-8 (CONTINUED)

08/14/75 AUTOFLOW CHART SET - FINAL -DECODE- FLOWCHARTS PAGE 05

CHART TITLE - SUBROUTINE DECODE(I,N,RESP,IN,IN2)

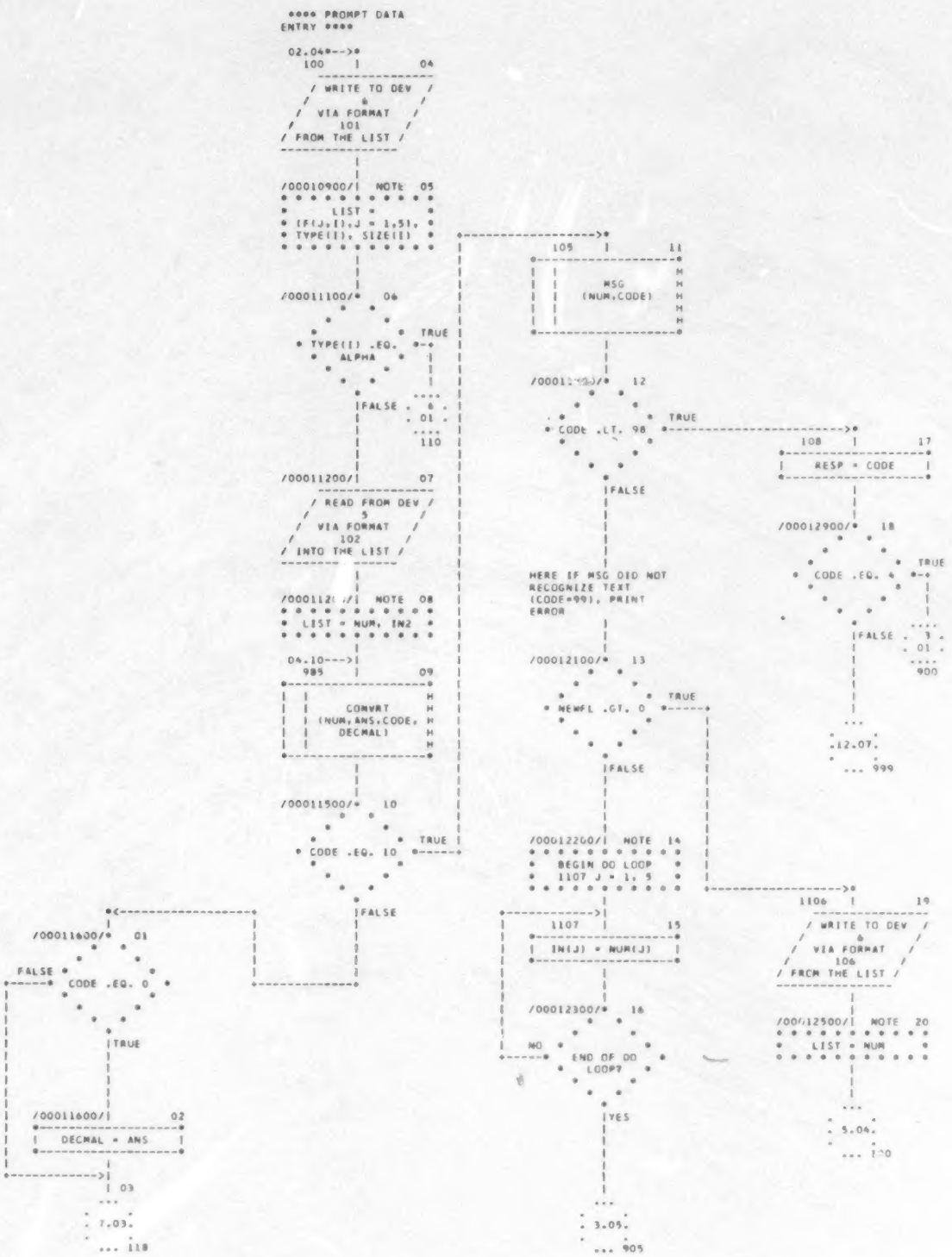


FIGURE V-8 (CONTINUED)

CHART TITLE - SUBROUTINE DECODE(I,M,RESP,IN,IN2)

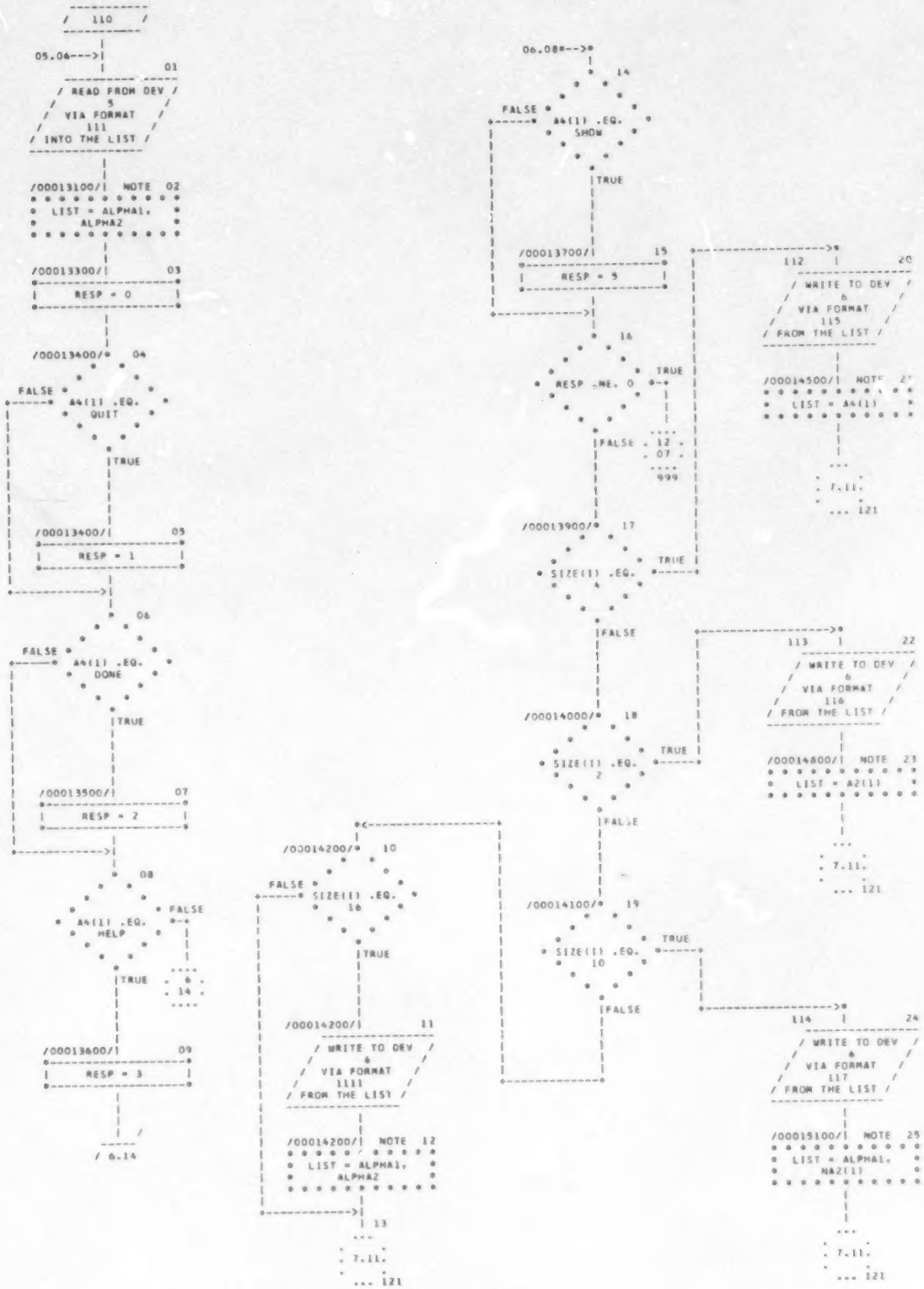


FIGURE V-8 (CONTINUED)

CHART TITLE - SUBROUTINE DECODE(I,N,RESP,IN,IN2)

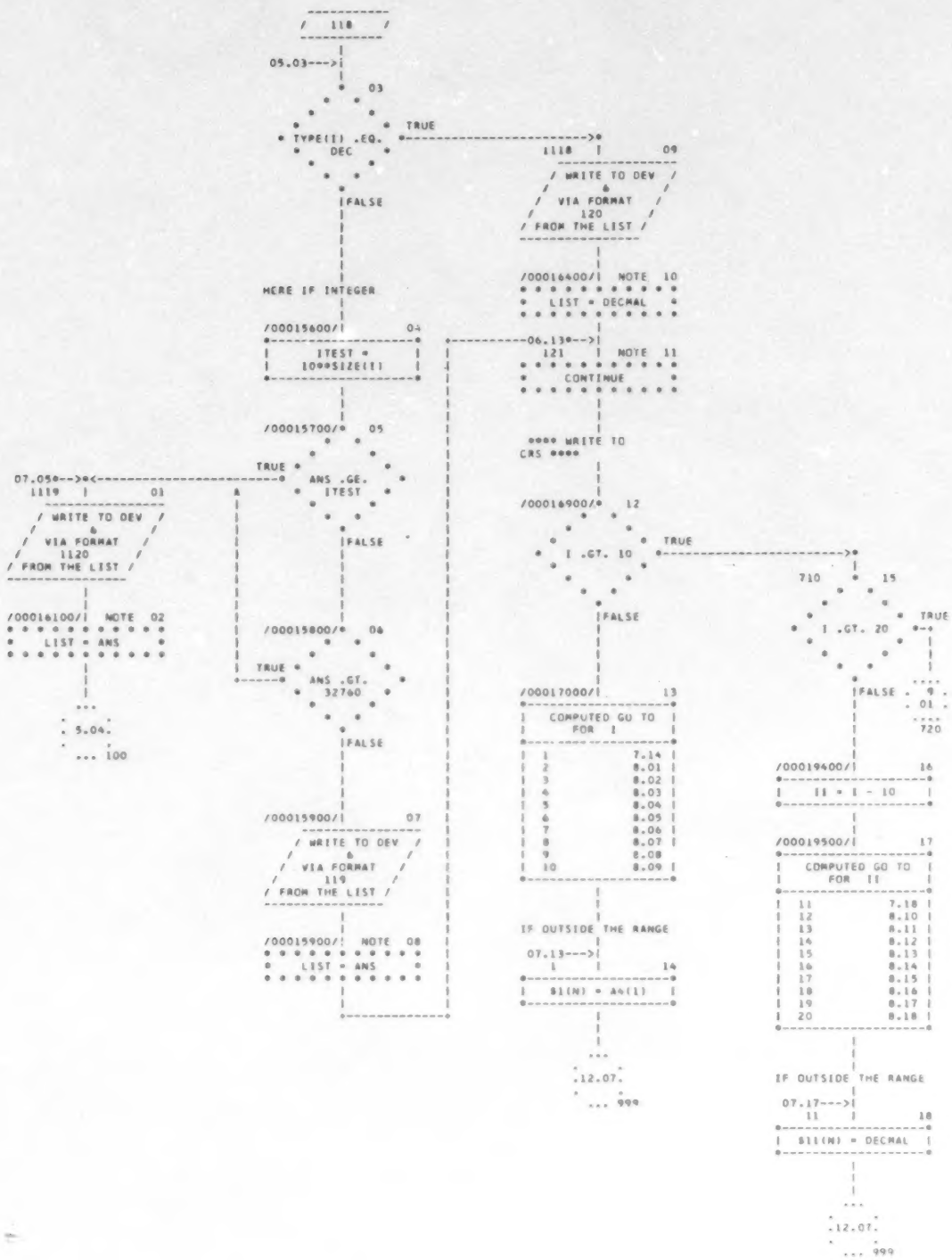


FIGURE V-8 (CONTINUED)

CHART TITLE - SUBROUTINE DECODE(I,N,RESP,IN,IN2)

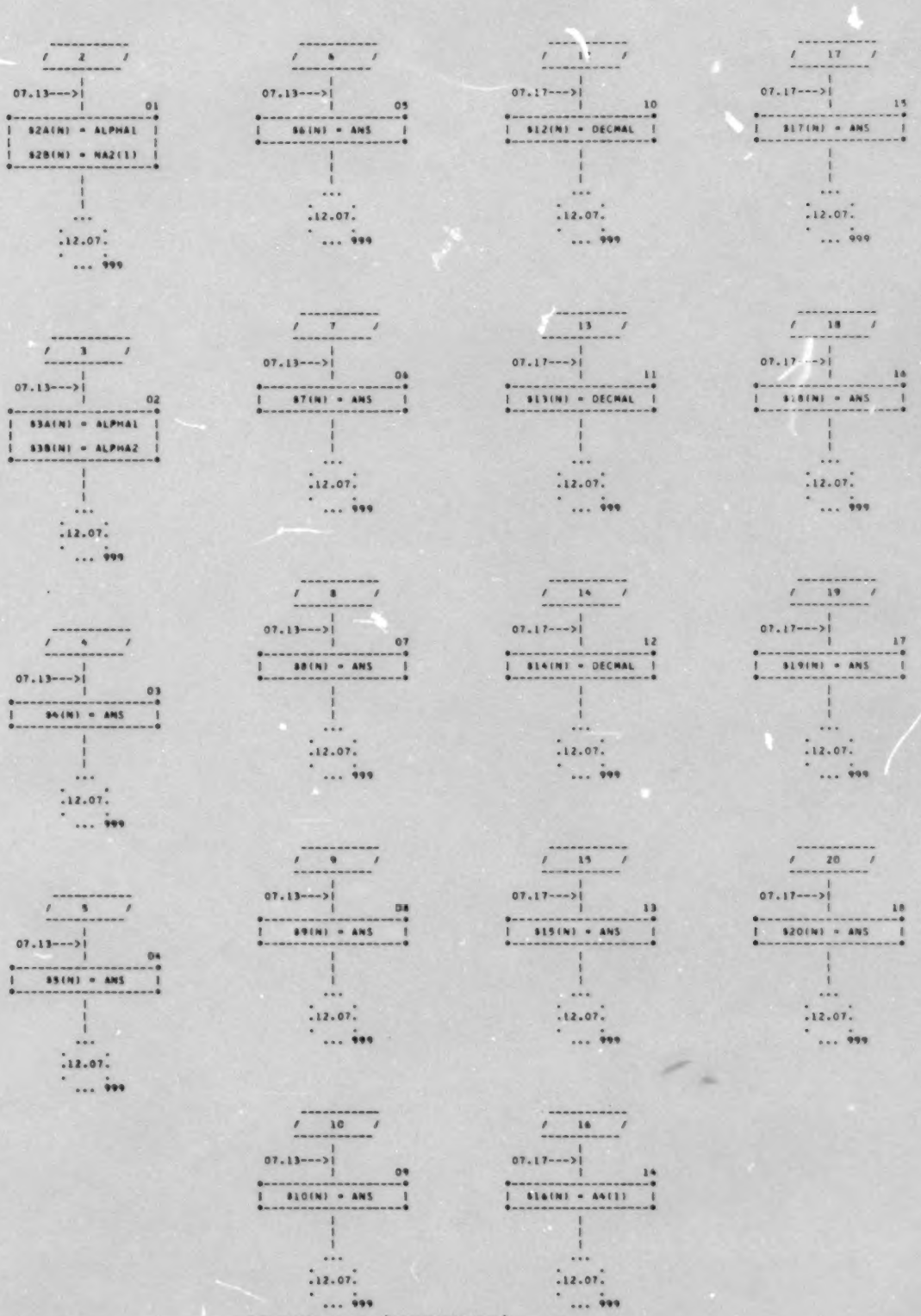


FIGURE V-8 (CONTINUED)

CHART TITLE - SUBROUTINE DECODE(I,...,RESP,IN,IN2)

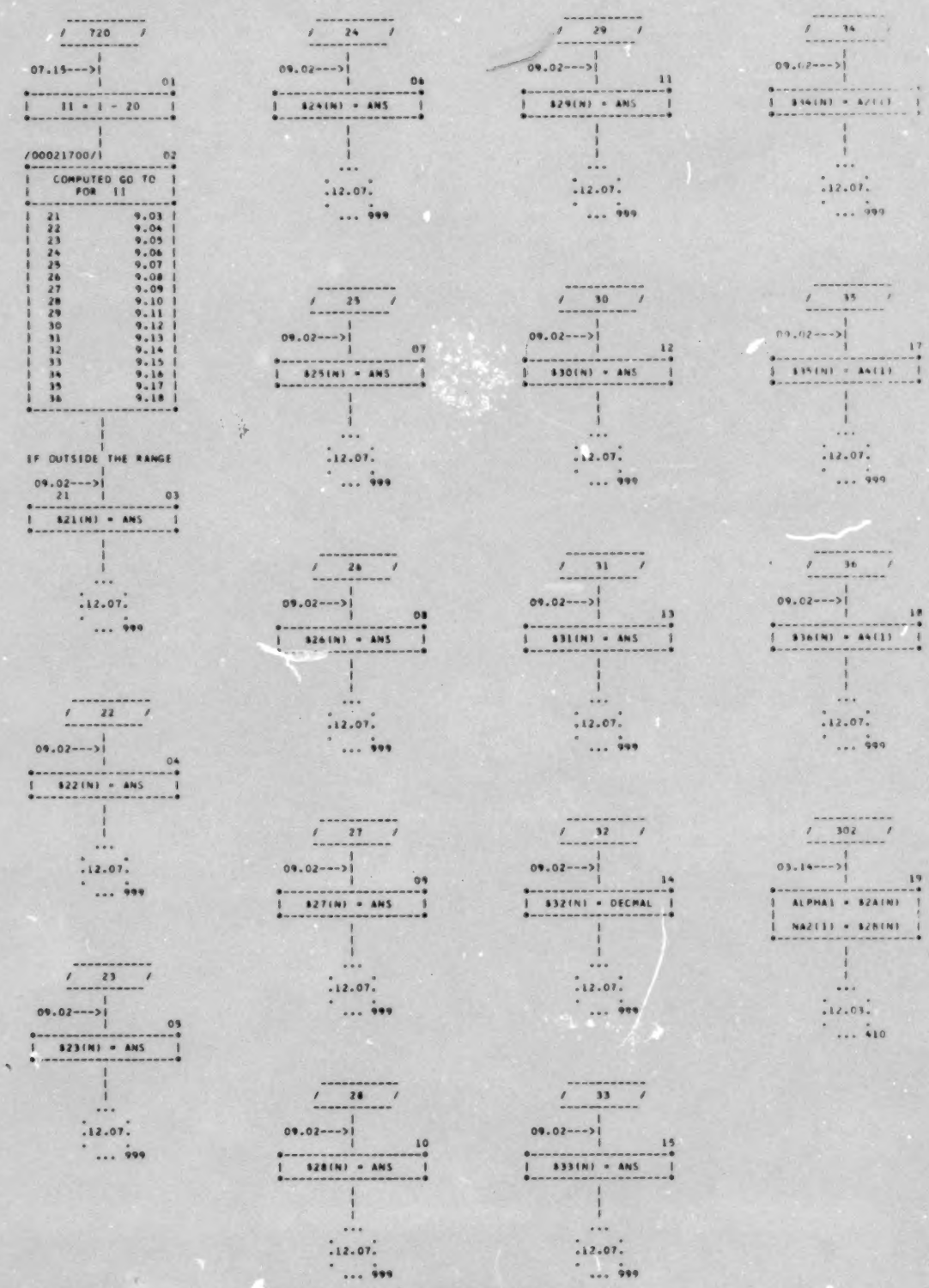


FIGURE V-8 (CONTINUED)

08/14/75 AUTOFLOW CHART SET - FINAL -DECODE- FLOWCHARTS PAGE

CHART TITLE - SUBROUTINE DECODE(I,N,RESP,IN,IN2)

```
 ----------
/   303    /
----------
|
03.14--->|
|        01
*------------------*
|  ALPHA1 = $3A(N)  |
|                   |
|  ALPHA2 = $3B(N)  |
*------------------*
|
|
...
.   .
.12.05.
.   .
... 416
```

```
 ----------
/   304    /
----------
|
03.14--->|
|        02
*------------------*
|   ANS = $4(N)     |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   305    /
----------
|
03.14--->|
|        03
*------------------*
|  DECMAL = $5(N)   |
*------------------*
|
|
...
.   .
.11.18.
.   .
... 401
```

```
 ----------
/   306    /
----------
|
03.14--->|
|        04
*------------------*
|   ANS = $6(N)     |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   307    /
----------
|
03.14--->|
|        05
*------------------*
|   ANS = $7(N)     |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   308    /
----------
|
03.14--->|
|        06
*------------------*
|   ANS = $8(N)     |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   309    /
----------
|
03.14--->|
|        07
*------------------*
|   ANS = $9(N)     |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   310    /
----------
|
03.14--->|
|        08
*------------------*
|  ANS = $10(N)     |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   312    /
----------
|
03.18--->|
|        09
*------------------*
|  DECMAL = $12(N)  |
*------------------*
|
|
...
.   .
.11.18.
.   .
... 401
```

```
 ----------
/   313    /
----------
|
03.18--->|
|        10
*------------------*
|  DECMAL = $13(N)  |
*------------------*
|
|
...
.   .
.11.18.
.   .
... 401
```

```
 ----------
/   314    /
----------
|
03.18--->|
|        11
*------------------*
|  DECMAL = $14(N)  |
*------------------*
|
|
...
.   .
.11.11.
.   .
... 401
```

```
 ----------
/   315    /
----------
|
03.18--->|
|        12
*------------------*
|   ANS = $15(N)    |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   316    /
----------
|
03.18--->|
|        13
*------------------*
|  A4(I) = $16(N)   |
*------------------*
|
|
...
.   .
.11.20.
.   .
... 404
```

```
 ----------
/   317    /
----------
|
03.18--->|
|        14
*------------------*
|   ANS = $17(N)    |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   318    /
----------
|
03.18--->|
|        15
*------------------*
|  DECMAL = $18(N)  |
*------------------*
|
|
...
.   .
.11.18.
.   .
... 401
```

```
 ----------
/   319    /
----------
|
03.18--->|
|        16
*------------------*
|   ANS = $19(N)    |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   320    /
----------
|
03.18--->|
|        17
*------------------*
|   ANS = $20(N)    |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

```
 ----------
/   220    /
----------
|
03.16--->|
|        18
*------------------*
|   II = I - 20     |
*------------------*
|
|
/000/9900/|        19
*------------------*
|  COMPUTED GO TO   |
|      FOR  II      |
*------------------*
|  321        10.20 |
|  322        11.01 |
|  323        11.02 |
|  324        11.03 |
|  325        11.04 |
|  326        11.05 |
|  327        11.06 |
|  328        11.07 |
|  329        11.08 |
|  330        11.09 |
|  331        11.10 |
|  332        11.11 |
|  333        11.12 |
|  334        11.13 |
|  335        11.14 |
|  336        11.15 |
*------------------*
|
|
IF OUTSIDE THE RANGE
|
10.19--->|
 321     |        20
*------------------*
|   ANS = $21(N)    |
*------------------*
|
|
...
.   .
.11.16.
.   .
... 400
```

FIGURE V-8 (CONTINUED)

CHART TITLE - SUBROUTINE DECODE(I,N,RESP,IN,IN2)

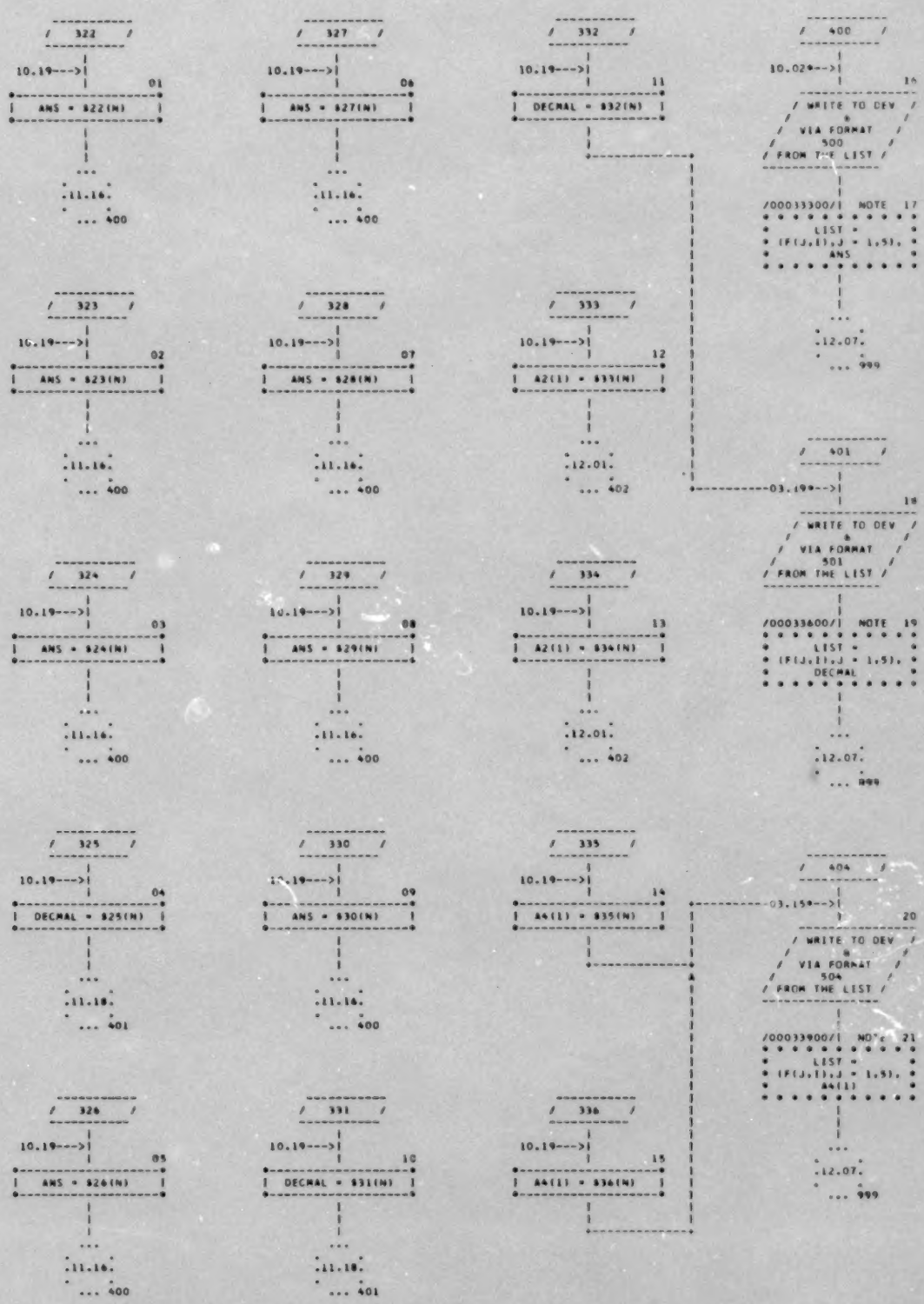


FIGURE V-8 (CONTINUED)

CHART TITLE - SUBROUTINE DECODE(I,N,RESP,IN,IN2)

```
                     ------------
                    /   402     /
                    ------------
                         |
            11.12*-->|
                         |          01
                    ----------------
                   / WRITE TO DEV  /
                  /       6       /
                 /  VIA FORMAT   /
                /      502      /
               / FROM THE LIST /
               ----------------
                         |
                         |
            /00034200/|  NOTE  02
            * * * * * * * * * * *
            *     LIST =          *
            * (F(J,I),J = 1,5),   *
            *      A2(I)          *
            * * * * * * * * * * *
                         |
                         |
         *---------------*
         |
         |
         |
         |                ------------
         |               /   410     /
         |               ------------
         |                    |
         |      09.19--->|
         |                    |          03
         |               ----------------
         |              / WRITE TO DEV  /
         |             /       6       /
         |            /  VIA FORMAT   /
         |           /      510      /
         |          / FROM THE LIST /
         |          ----------------
         |                    |
         |                    |
         |       /00034500/|  NOTE  04
         |       * * * * * * * * * * *
         |       *     LIST =          *
         |       * (F(J,I),J = 1,5),   *
         |       *  ALPHA1, NA2(I)     *
         |       * * * * * * * * * * *
         |                    |
         |<-------------------*
         |
         |
         |
         |                ------------
         |               /   416     /
         |               ------------
         |                    |
         |      10.01--->|
         |                    |          05
         |               ----------------
         |              / WRITE TO DEV  /
         |             /       6       /
         |            /  VIA FORMAT   /
         |           /      516      /
         |          / FROM THE LIST /
         |          ----------------
         |                    |
         |                    |
         |       /00034800/|  NOTE  06
         |       * * * * * * * * * * *
         |       *     LIST =          *
         |       * (F(J,I),J = 1,5),   *
         |       *  ALPHA1, ALPHA2     *
         |       * * * * * * * * * * *
         |                    |
         *----02.10*-->|
                   999        | 07
                         ---------
                         * EXIT  *
                         ---------
```

FIGURE V-8 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
IMPLICIT INTEGER*2 (B)
INTEGER*4 B1,B16,B35,B36
REAL*4 B5,B11,B12,B13,B14,B18,B25,B31,B32
REAL*8 B2A,B3A,B3B
COMMON /A/ B1(144),B2A(144),B3A(144),B4(144),B5(144),B6(144)
COMMON /A/ B7(144),B8(144),B9(144),B10(144),B11(144),B12(144)
COMMON /A/ B13(144),B14(144),B15(144),B16(144),B17(144),B18(144)
COMMON /A/ B19(144),B20(144),B21(144),B22(144),B23(144),B24(144)
COMMON /A/ B25(144),B26(144),B27(144),B28(144),B29(144),B30(144)
COMMON /A/ B31(144),B32(144),B33(144),B34(144),B35(144),B36(144)
COMMON /A/ B2B (144),B3B(144)
INTEGER*2 NUM (5) /5*0/, CODE, TYPE (36), DEC, INT, ALPHA
DATA TYPE /3*'A', 7*'I', 4*'F', 'I', 'A', 15*'I','F', 4*'A'/
DATA ALPHA /'A'/, INT /'I'/, DEC /'F'/
REAL*8 ALPHA1, ALPHA2
INTEGER*4 ANS, A4 (2),QUIT/'QUIT'/,DONE/'DONE'/,HELP/'HELP'/,SHOW/
'SHOW'/
INTEGER*2 IN (5),IN2(5),F (5,36), CDP (5), CTLG (5), CRS (5)
INTEGER*2 CONV1 (5), LEN1 (5), CAP1 (5), BCAP1 (5), DMD1 (5)
INTEGER*2 BDMD1 (5), OFST (5), PFAIL (5), NADIS (5), NOSHO (5)
INTEGER*2 SETB (5), BKLG (5), JNO (5), CONV2 (5), LEN2 (5)
INTEGER*2 CAP2 (5), BCAP2 (5), DMD2 (5), BDMD2 (5), CHWK2 (5)
INTEGER*2 CONV3 (5), LEN3 (5), CAP3 (5), BCAP3 (5), DMD3 (5)
INTEGER*2 BDMD3 (5), CHWK3 (5), PAOB (5), AFAIL (5), PRTY (5)
INTEGER*2 CTYPE (5), PNEC (5), SCH (5), BLANK /' '/
INTEGER*2 RESP, SIZE (36), A2 (4), NA2 (4)
DATA SIZE /4, 10, 16, 3, 4, 2*3, 2*5, 2, 4*5, 2*4, 3, 4,
 2*3, 2*5, 2*3, 4, 2*3, 2*5, 3, 4, 5, 2*2, 2*4/
EQUIVALENCE (A2,ALPHA1), (A4,ALPHA1), (NA2,ALPHA2)
EQUIVALENCE (CDP, F(1,1)), (CTLG, F(1,2)), (CRS, F(1,3))
EQUIVALENCE (CONV1, F(1,4)), (LEN1, F(1,5)), (CAP1, F(1,6))
EQUIVALENCE (BCAP1, F(1,7)), (DMD1, F(1,8)), (BDMD1, F(1,9))
EQUIVALENCE (OFST, F(1,10)), (PFAIL, F(1,11)), (NADIS, F(1,12))
EQUIVALENCE (NOSHO, F(1,13)), (SETB, F(1,14)), (BKLG, F(1,15))
EQUIVALENCE (JNO, F(1,16)), (CONV2, F(1,17)), (LEN2, F(1,18))
EQUIVALENCE (CAP2, F(1,19)), (BCAP2, F(1,20)), (DMD2, F(1,21))
EQUIVALENCE (BDMD2, F(1,22)), (CHWK2, F(1,23)), (CONV3, F(1,24))
EQUIVALENCE (LEN3, F(1,25)), (CAP3, F(1,26)), (BCAP3, F(1,27))
EQUIVALENCE (DMD3, F(1,28)), (BDMD3, F(1,29)), (CHWK3, F(1,30))
EQUIVALENCE (PAOB, F(1,31)), (AFAIL, F(1,32)), (PRTY, F(1,33))
EQUIVALENCE (CTYPE, F(1,34)), (PNEC, F(1,35)), (SCH, F(1,36))
DATA CDP/'C','D','P',' ',' '/,CTLG/'C','T','L','G',' '/
DATA CRS/'C','R','S',' ',' '/,CONV1/'C','O','N','V','1'/
DATA LEN1/'L','E','N','1',' '/,CAP1/'C','A','P','1',' '/
DATA BCAP1/'B','C','A','P','1'/,DMD1/'D','M','D','1',' '/
DATA BDMD1/'B','D','M','D','1'/,OFST/'O','F','S','T',' '/
DATA PFAIL/'P','F','A','I','L'/,NADIS/'N','A','D','I','S'/
DATA NOSHO/'N','O','S','H','O'/,SETB/'S','E','T','B',' '/
```

FIGURE V-8 (CONTINUED)

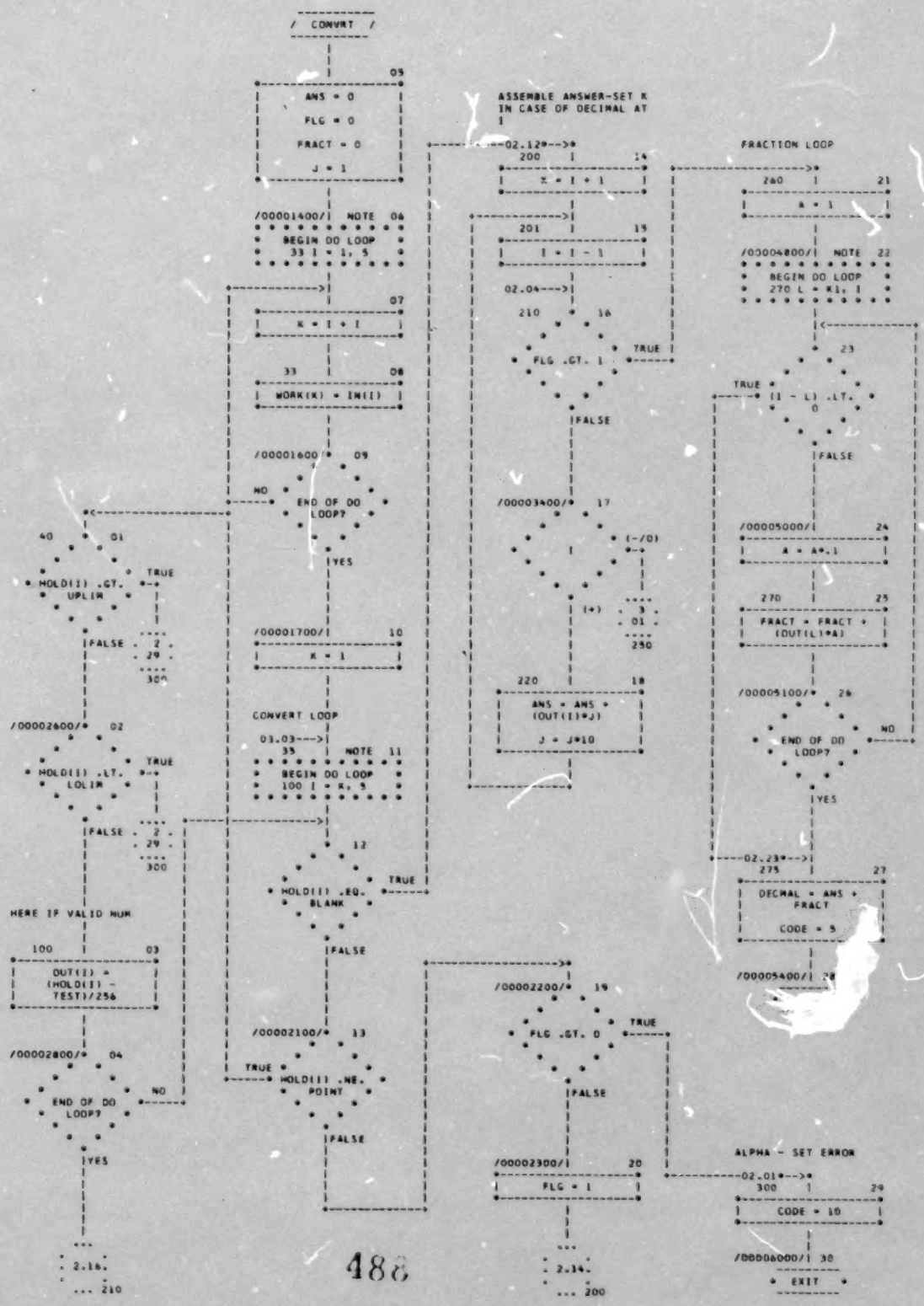
CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      DATA SCH/'S','C','H',' ',' '/,BKLG/'B','K','L','G',' '/
      DATA JNO/'J','N','O',' ',' '/,CONV2/'C','O','N','V','2'/
      DATA LEN2/'L','E','N','2',' '/,CAP2/'C','A','P','2',' '/
      DATA BCAP2/'B','C','A','P','2'/,DMD2/'D','M','D','2',' '/
      DATA BDMD2/'B','D','M','D','2'/,CHWK2/'C','H','W','K','2'/
      DATA CONV3/'C','O','N','V','3'/,LEN3/'L','E','N','3',' '/
      DATA CAP3/'C','A','P','3',' '/,BCAP3/'B','C','A','P','3'/
      DATA DMD3/'D','M','D','3',' '/,BDMD3/'B','D','M','D','3'/
      DATA CHWK3/'C','H','W','K','3'/,PAOB/'P','A','O','B',' '/
      DATA AFAIL/'A','F','A','I','L'/,PRTY/'P','R','T','Y',' '/
      DATA CTYPE/'C','T','Y','P','E'/,PNEC/'P','N','E','C',' '/
101   FORMAT (' +ENTER ',5A1, 2X, '[',A1, I2, ']')
102   FORMAT (10A1)
106   FORMAT (1X, 5A1, ' +INVALID- RETRY'/)
111   FORMAT (2A8)
1111  FORMAT (1X,2A8)
115   FORMAT (1X,A4)
116   FORMAT (1X,A2)
117   FORMAT (1X, A8, A2)
1120  FORMAT (1X,I5,' TOO LARGE- RETRY'/)
119   FORMAT (1X, I5)
120   FORMAT (F9.3)
500   FORMAT (/2X,5A1,1X,I5/)
501   FORMAT (/2X,5A1,F9.3/)
504   FORMAT (/2X,5A1,2X,A4/)
502   FORMAT (/2X,5A1,A2,/)
510   FORMAT (/2X,5A1,A8,A2/)
516   FORMAT (/2X,5A1,A8,A8/)
```

FIGURE V-8 (CONTINUED)

CHART TITLE - SUBROUTINE CONVRT(IN,ANS,CODE,DECMAL)

AUTOFLOW - SUBROUTINE CONVRT



488

FIGURE V-9

CHART TITLE - SUBROUTINE CONVRT(IN,ANS,CODE,DECMAL)

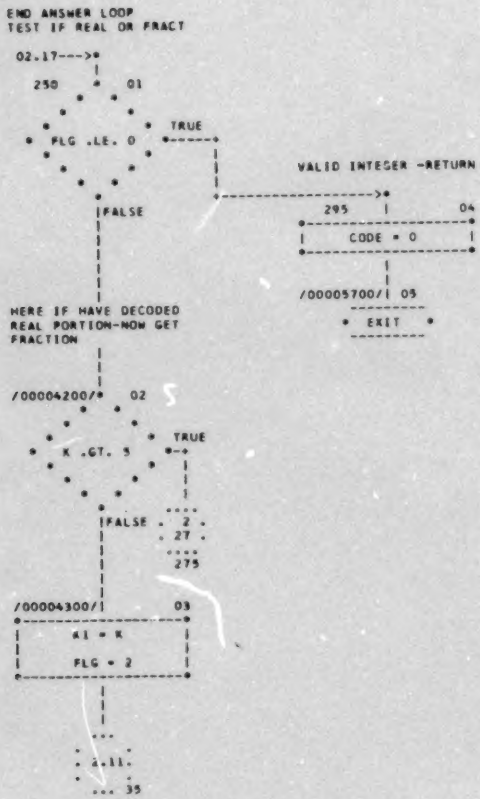


CHART TITLE - NON-PROCEDURAL STATEMENTS

```
INTEGER*2 IN (5) , WORK (10), CODE
INTEGER*4 UPLIM, LOLIM, BLANK, HOLD (5), TEST, OUT (5), ANS,POINT
EQUIVALENCE (HOLD, WORK)
DATA WORK/10*0/,TEST/ZF000/,UPLIM/ZF940/,LOLIM/ZF040/,BLANK/Z4040/
DATA POINT /Z4B40/
```

FIGURE V-9 (CONTINUED)

CHART TITLE - SUBROUTINE MSG(IN,CODE)

AUTOFLOW - SUBROUTINE MSG

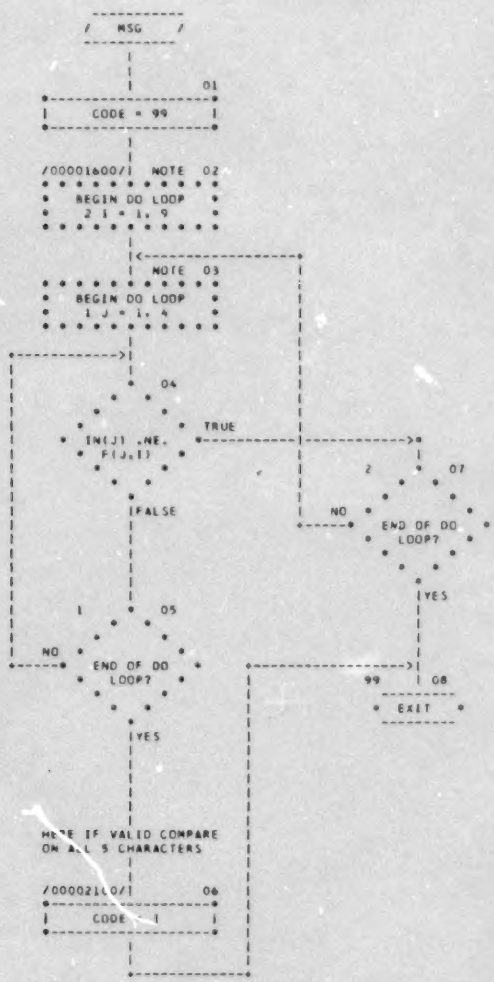


FIGURE V-10

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
INTEGER*2 F(4,9), IN(5), CODE, QUIT(4), DONE(4), HELP(4), STAT(4)
INTEGER*2 SHOW (4),  COPY (4), NEW (4), ALL (4), MOD (4)
EQUIVALENCE (QUIT,F(1,1)),(DONE,F(1,2)),(HELP,F(1,3)),(NEW,F(1,7))
EQUIVALENCE (SHOW,F(1,5)),(COPY,F(1,6)),(ALL,F(1,8)),(MOD,F(1,9))
EQUIVALENCE (STAT,F(1,4))
DATA QUIT /'Q','U','I','T'/, DONE /'D','O','N','E'/
DATA HELP /'H','E','L','P'/, SHOW /'S','H','O','W'/
DATA COPY /'C','O','P','Y'/, NEW  /'N','E','W',' '/
DATA ALL  /'A','L','L',' '/, MOD  /'M','O','D',' '/
DATA STAT /'S','T','A','T'/
```

FIGURE V-10 (CONTINUED)

CHART TITLE - SUBROUTINE SHOW(N)

AUTOFLOW - SUBROUTINE SHOW

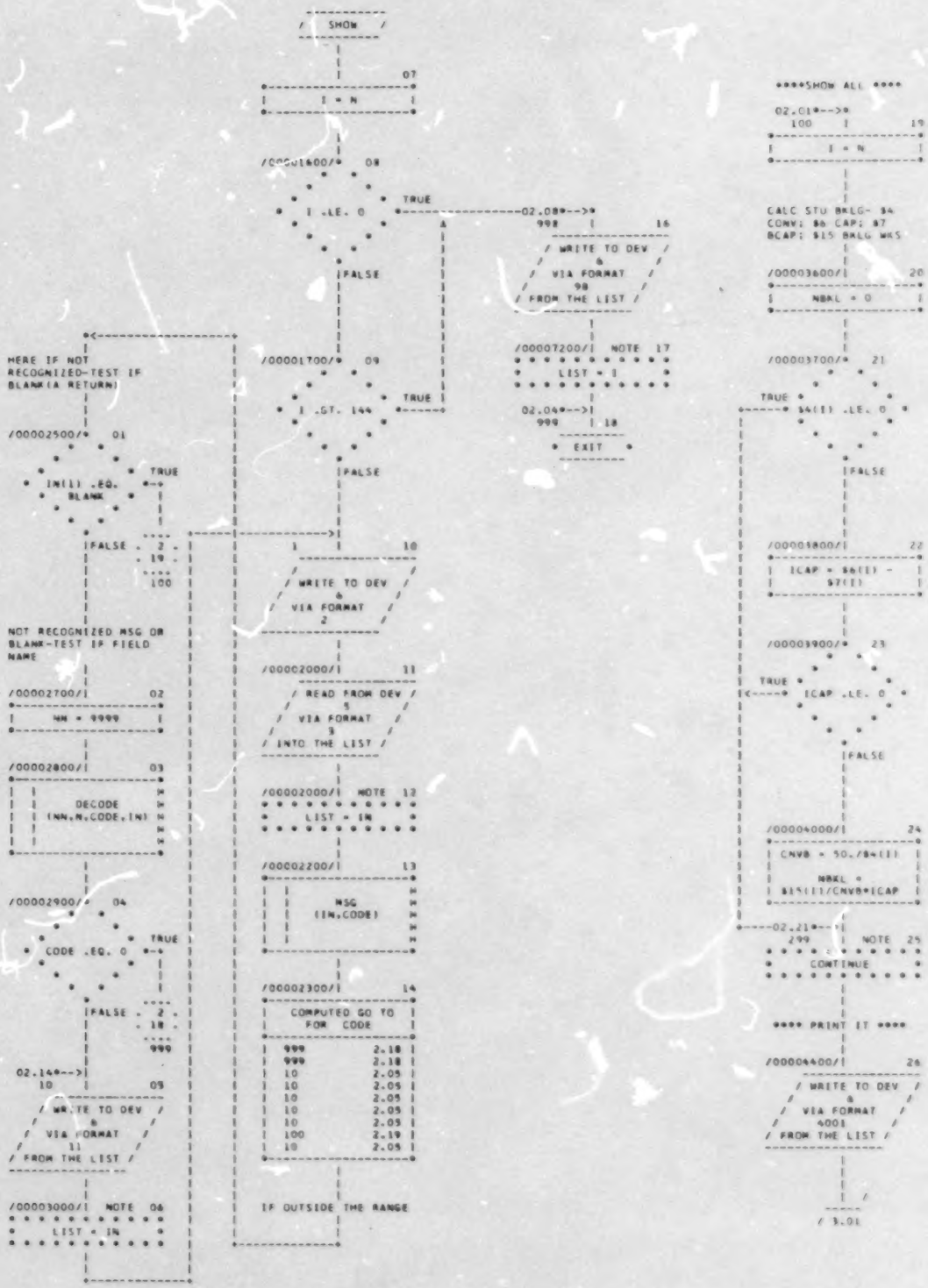


FIGURE V-11

CHART TITLE - SUBROUTINE SHOW(N)

```
 02.26--->*
/00004400/|  NOTE  01
* * * * * * * * * * *
*   LIST = $1(I),   *
*  $3A(I), $3B(I)   *
* * * * * * * * * * *
          |
          |
/00004500/|        02
 ----------------
   / WRITE TO DEV  /
  /       &       /
 /  VIA FORMAT   /
/     4002      /
/ FROM THE LIST /
----------------
          |
          |
/00004500/|  NOTE  03
* * * * * * * * * * *
*   LIST = $4(I),   *
*  $17(I), $24(I),  *
*       NBKL        *
* * * * * * * * * * *
          |
          |
/00004600/|        04
 ----------------
   / WRITE TO DEV  /
  /       &       /
 /  VIA FORMAT   /
/     4003      /
/ FROM THE LIST /
----------------
          |
          |
/00004600/|  NOTE  05
* * * * * * * * * * *
*   LIST = $5(I),   *
*  $18(I), $25(I),  *
*       $15(I)      *
* * * * * * * * * * *
          |
          |
/00004700/|        06
 ----------------
   / WRITE TO DEV  /
  /       &       /
 /  VIA FORMAT   /
/     4004      /
/ FROM THE LIST /
----------------
          |
          |
/00004700/|  NOTE  07
* * * * * * * * * * *
*  LIST = $36(I),   *
*  $16(I), $2A(I),  *
*  $28(I), $35(I),  *
*  $6(I), $19(I),   *
*      $26(I)       *
* * * * * * * * * * *
          |
          |
/00004900/|        08
 ----------------
   / WRITE TO DEV  /
  /       &       /
 /  VIA FORMAT   /
/     4005      /
/ FROM THE LIST /
----------------
          |
          |
/00004900/|  NOTE  09
* * * * * * * * * * *
*   LIST = $7(I),   *
*  $20(I), $27(I),  *
*      $23(I)       *
* * * * * * * * * * *
          |
          |
/00005000/|        10
 ----------------
   / WRITE TO DEV  /
  /       &       /
 /  VIA FORMAT   /
/     4006      /
/ FROM THE LIST /
----------------
          |
          |
          *--------------*
                         |
                         |
*------------------->*   |
|        /00005000/|  NOTE  11
|        * * * * * * * * * * *
|        *   LIST = $8(I),   *
|        *  $21(I), $28(I),  *
|        *      $30(I)       *
|        * * * * * * * * * * *
|                  |
|                  |
|        /00005100/|        12
|         ----------------
|           / WRITE TO DEV  /
|          /       &       /
|         /  VIA FORMAT   /
|        /     4007      /
|        / FROM THE LIST /
|        ----------------
|                  |
|                  |
|        /00005100/|  NOTE  13
|        * * * * * * * * * * *
|        *  LIST = $31(I),   *
|        *  $32(I), $34(I),  *
|        *  $33(I), $9(I),   *
|        *  $22(I), $29(I),  *
|        *      $10(I)       *
|        * * * * * * * * * * *
|                  |
|                  |
|        /00005300/|        14
|        *-------------------*
|        |      RATEA =     |
|        |    $11(I)*100.   |
|        |                  |
|        |      RATEB =     |
|        |    $12(I)*100.   |
|        |                  |
|        |      RATEC =     |
|        |    $13(I)*100.   |
|        *-------------------*
|                  |
|                  |
|        /00005600/|        15
|         ----------------
|           / WRITE TO DEV  /
|          /       &       /
|         /  VIA FORMAT   /
|        /     4008      /
|        / FROM THE LIST /
|        ----------------
|                  |
|                  |
|        /00005600/|  NOTE  16
|        * * * * * * * * * * *
|        *   LIST = RATEA,   *
|        *   RATEB, RATEC    *
|        * * * * * * * * * * *
|                  |
|                  |
|                  |
|                 ...
|                .   .
|                . 2.18.
|                .   .
|                 ... 999
```

FIGURE V-11 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      IMPLICIT INTEGER*2 ($)
      INTEGER*4 $1,$16,$35,$36
      REAL*4 $5,$11,$12,$13,$14,$18,$25,$31,$32
      REAL*8 $2A,$3A,$38
      COMMON /A/ $1(144),$2A(144),$3A(144),$4(144),$5(144),$6(144)
      COMMON /A/ $7(144),$8(144),$9(144),$10(144),$11(144),$12(144)
      COMMON /A/ $13(144),$14(144),$15(144),$16(144),$17(144),$18(144)
      COMMON /A/ $19(144),$20(144),$21(144),$22(144),$23(144),$24(144)
      COMMON /A/ $25(144),$26(144),$27(144),$28(144),$29(144),$30(144)
      COMMON /A/ $31(144),$32(144),$33(144),$34(144),$35(144),$36(144)
      COMMON /A/ $28 (144),$38(144)
      INTEGER*2 IN (5) /5*0/, BLANK /' '/, CODE
2     FORMAT (/' SHOW: ENTER ALL /A FIELD NAME')
3     FORMAT (5A1)
11    FORMAT (1X,5A1,' *INVALID- RETRY'/)
4001  FORMAT (//1X,A4, 5X, 2A8, 4X, '--SCHEDULE DATA--   FY   FY+1   FY+
     2   CRS BACKLOG')
4002  FORMAT (' ', T31, 'ANNUAL CONVENENGS', I5, 2I7, 4X, '# STU', I5)
4003  FORMAT (' SCHL JNO  CATALOG NO  NEC    COURSE LENGTH DYS', F5.1,
      2F7.1, 4X, '# WKS', I5)
4004  FORMAT (' ', 2 (A4, 1X), A8, A2, 1X, A4, 4X, 'COURSE CAPACITY',
      3I7)
4005  FORMAT (' ', T31, 'BUPERS CAPACITY', 3I7, 3X, 'FY+1 WK', I4)
4006  FORMAT (' P AOB  PFAIL  TYPE   PRTY    ANNUAL DEMAND', 2X, 3I7,
      3X, 'FY+2 WK', I4)
4007  FORMAT (' ', F5.1, F6.1, '%', 3X,A2,5X,A2,5X, 'BUPERS DEMAND',
      2X, 3I7, 3X, 'OFFSET', I5)
4008  FORMAT (/' RATES: FAIL',F5.1,'%  NADIS',F5.1,'%  NOSHOW',F5.1,'%
      '///)
98    FORMAT (/1X,I3,' COURSE NOT IN PROCESS- SHOW SKIPPED'/)
```

FIGURE V-11 (CONTINUED)

08/14/75 AUTOFLOW CHART SET - FINAL -SHOW- FLOWCHARTS PAGE 05

CHART TITLE - BLOCK DATA

```
 -----------
//00009200//
-----------
     |
     |
     | 01
 ---------
*  EXIT   *
 ---------

  RETURN
```

08/14/75 AUTOFLOW CHART SET - FINAL -SHOW- FLOWCHARTS PAGE 06

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
IMPLICIT INTEGER*2 ($)
INTEGER*4 $1,$16,$35,$36
REAL*4 $5,$11,$12,$13,$14,$18,$25,$31,$32
REAL*8 $2A,$3A,$3B
COMMON /A/ $1(144),$2A(144),$3A(144),$4(144),$5(144),$6(144)
COMMON /A/ $7(144),$8(144),$9(144),$10(144),$11(144),$12(144)
COMMON /A/ $13(144),$14(144),$15(144),$16(144),$17(144),$18(144)
COMMON /A/ $19(144),$20(144),$21(144),$22(144),$23(144),$24(144)
COMMON /A/ $25(144),$26(144),$27(144),$28(144),$29(144),$30(144)
COMMON /A/ $31(144),$32(144),$33(144),$34(144),$35(144),$36(144)
COMMON /A/ $2B (144),$3B(144)
DATA $1,$16,$35,$36/576*' '/,$2A,$3A,$3B/432*' '/,$2B/' '/
DATA $5,$11,$12,$13,$14,$18,$25,$31,$32 /1296*0./
DATA $4,$6,$7,$8,$[illegible],$10,$15,$17,$19,$20,$21,$22,$23,$24,$26,
 $27, $28, $29, $30,  $34 /2880 * 0/, $33 /' '/
```

FIGURE V-11 (CONTINUED)

FLOW - EXECUTIVE TPF

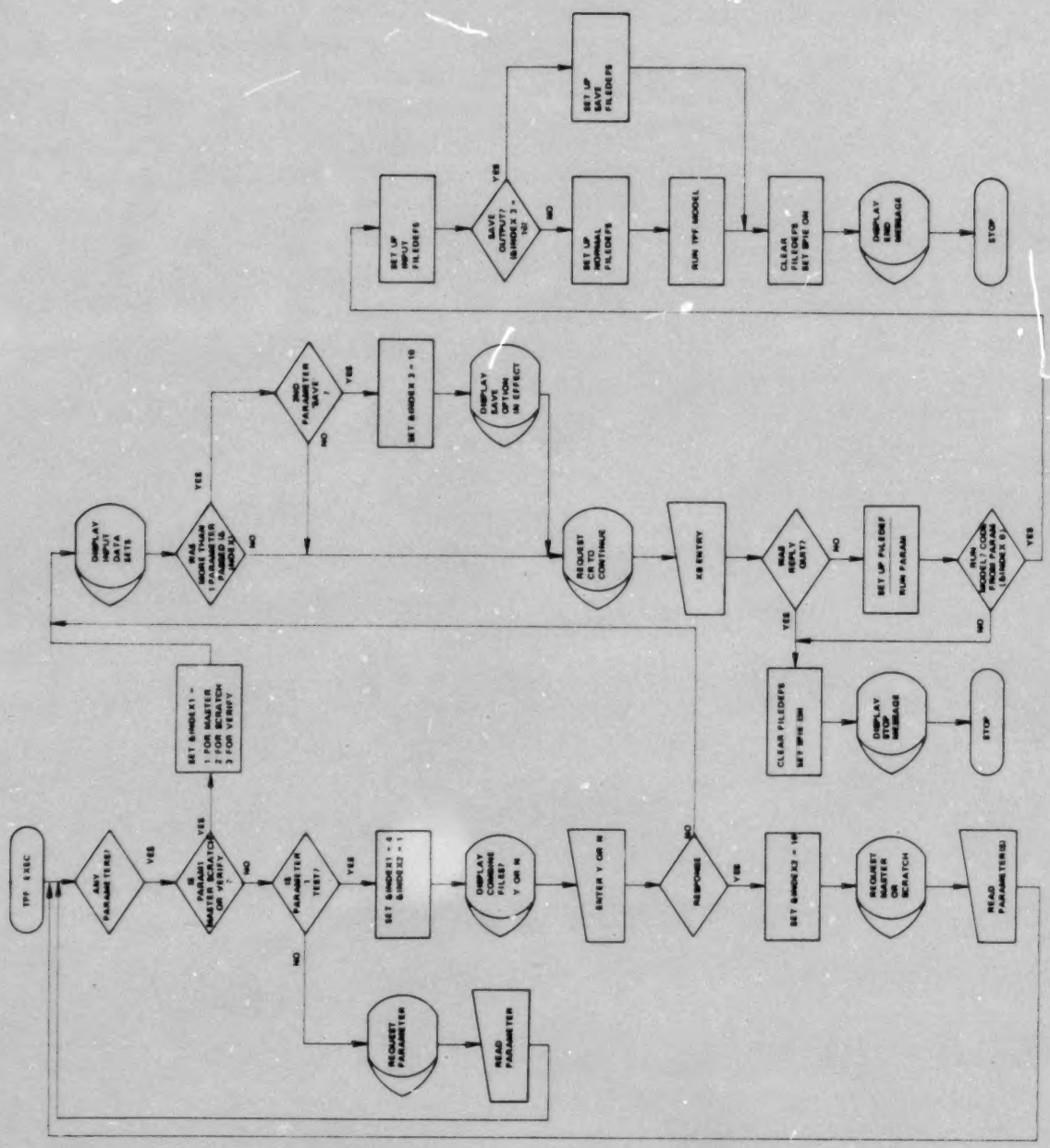


FIGURE V-12

CHART TITLE - PROC PARAM OPTIONS (MAIN)

AUTOFLOW - PROGRAM PARAM

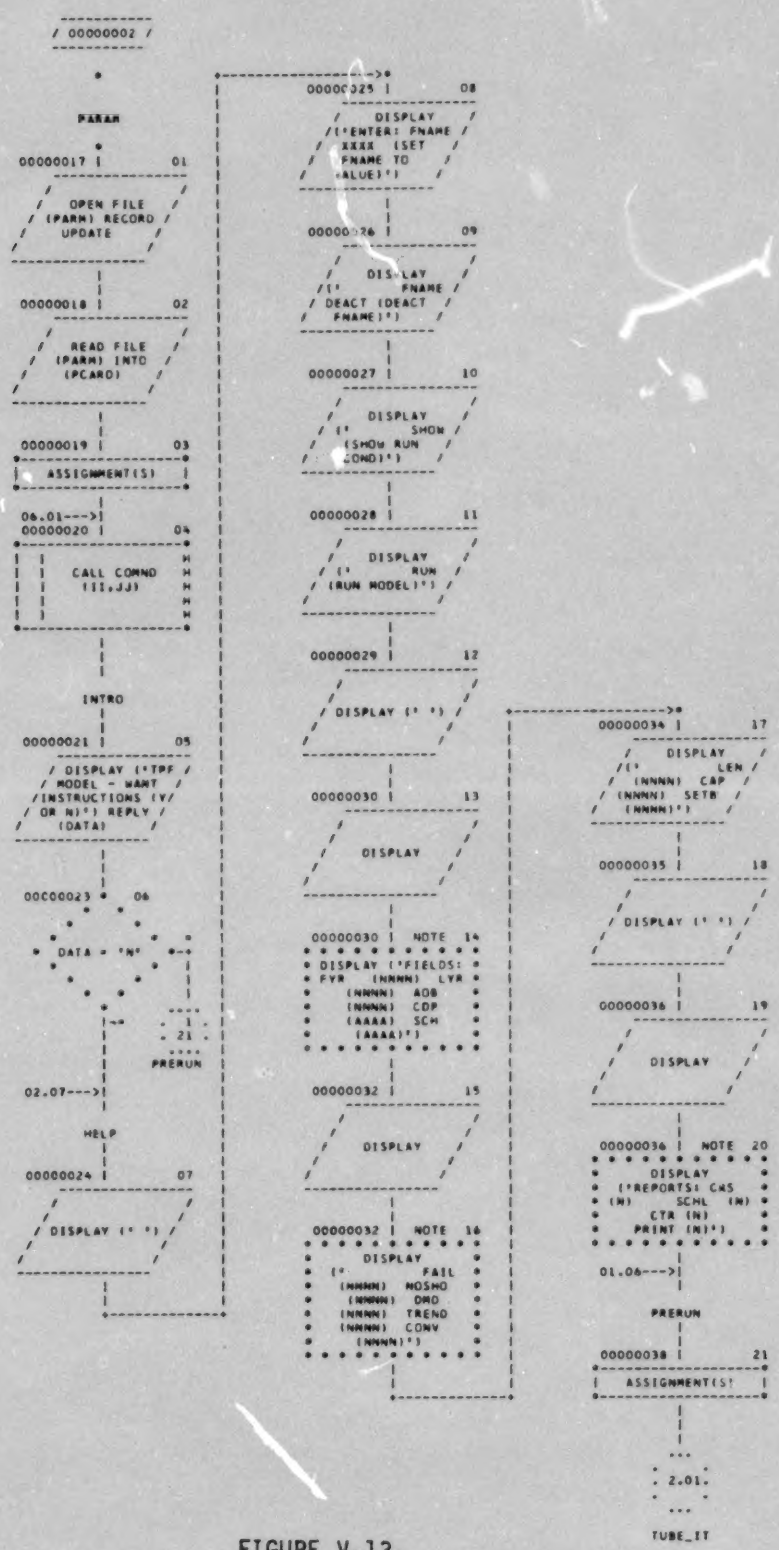


FIGURE V-13

CHART TITLE - PROC PARAM OPTIONS (MAIN)

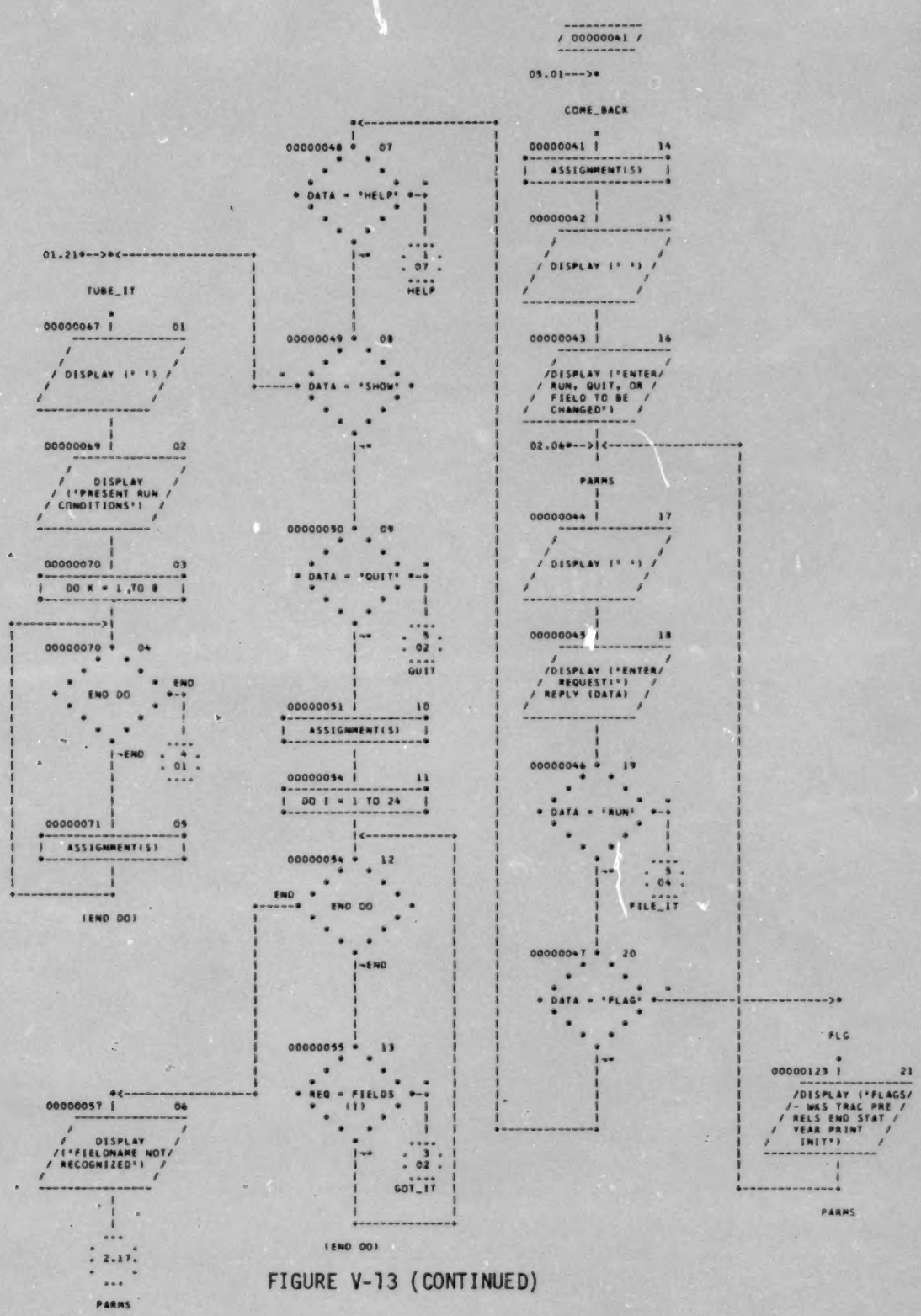


FIGURE V-13 (CONTINUED)

CHART TITLE - PROC PARAM OPTIONS (MAIN)

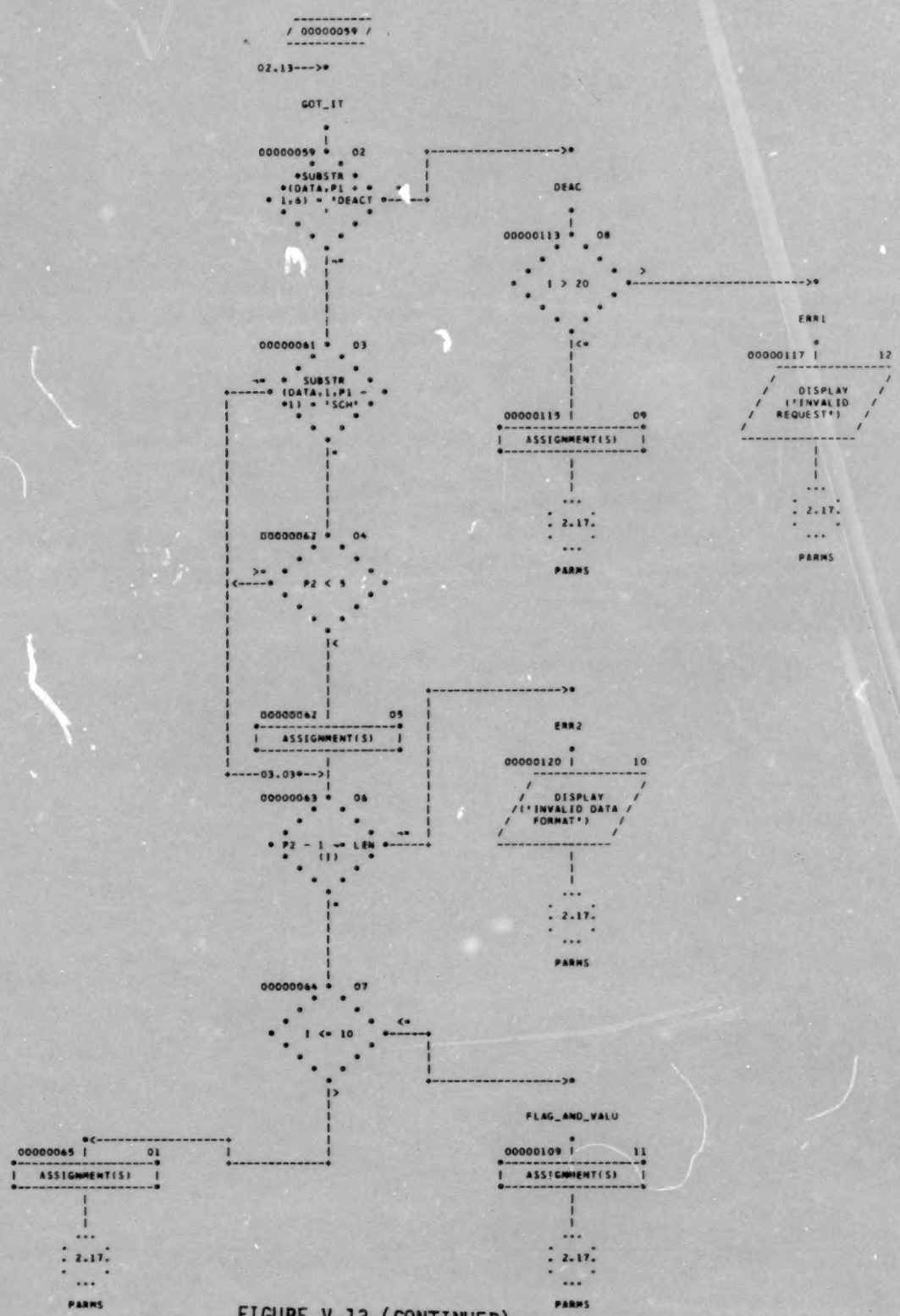


FIGURE V-13 (CONTINUED)

CHART TITLE - PROC PARAM OPTIONS (MAIN)

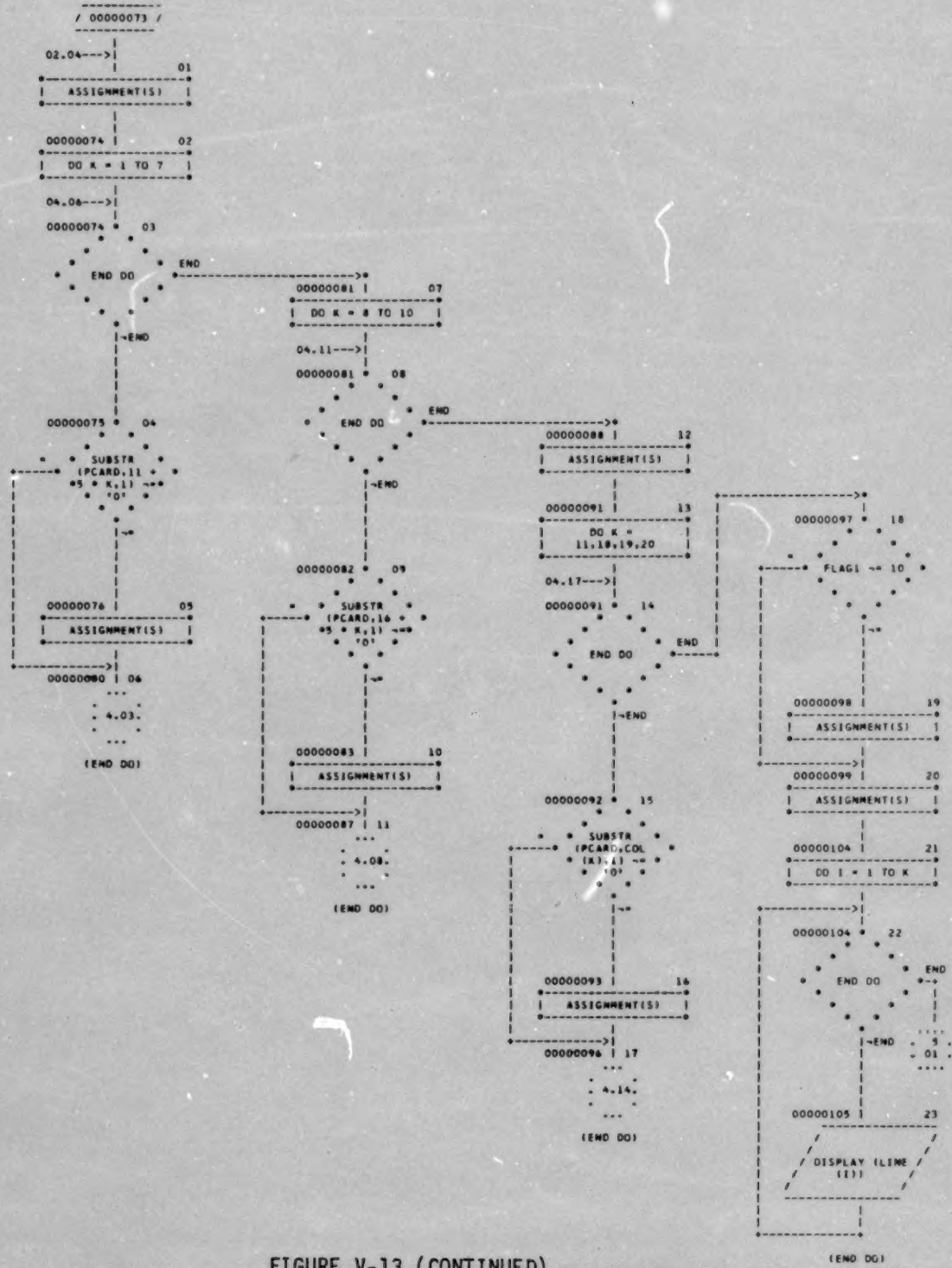
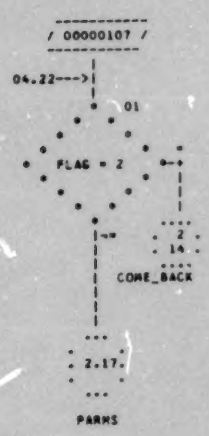


FIGURE V-13 (CONTINUED)

CHART TITLE - PROC PARAM OPTIONS (MAIN)



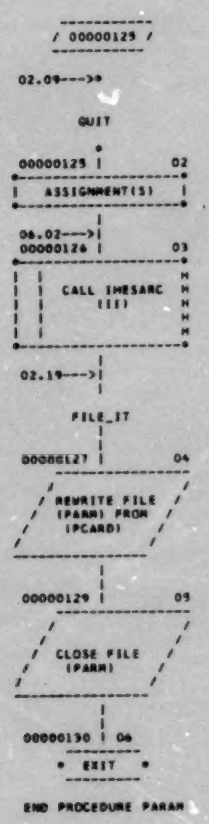


FIGURE V-13 (CONTINUED)

CHART TITLE - CALLED PROCEDURES - CROSS REFERENCE

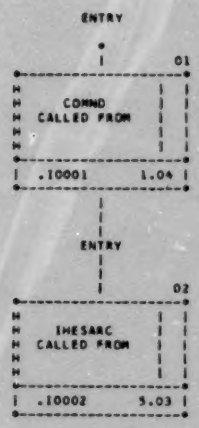


FIGURE V-13 (CONTINUED)

LISTING - EXECUTIVE TPFMOD

```
&TYPE OFF
-FRST &IF &INDEX EQ Ø &GOTO -NONE
&IF &1 EQ MASTER &GOTO -OK
&IF &1 EQ SCRATCH &GOTO -OK
&IF &1 EQ QUIT &EXIT
-NONE &PRINT REPLY MASTER OR SCRATCH (INPUT FILE)
&READ ARGS
&GOTO -FRST
-OK FILEDEF * CLEAR
FILEDEF 3 DSK TPF MASTER
FILEDEF 4 DSK TST DATA
&IF &1 NE SCRATCH &GOTO -MASTER
FILEDEF 3 DSK TPF SCRATCH
&PRINT SCRATCH DATA BASE USED AS MOD INPUT
&GOTO -RUNIT
-MASTER &PRINT MASTER DATA BASE USED AS MOD INPUT
-RUNIT TPFMOD1
FILEDEF * CLEAR
SET SPIE ON
&IF &INDEXØ GE 999 &GOTO -QUIT
-READ &PRINT RUN TPF MODEL? (Y OR N)
&READ ARGS
&IF &1 EQ Y &GOTO -MODEL
&IF &1 EQ N &GOTO -SKIP
&IF &1 EQ QUIT &GOTO -SKIP
&GOTO -READ
-MODEL EXEC TPF TEST
-SKIP &EXIT
-QUIT &PRINT NOT SAVED AT OPERATORS REQUEST
&EXIT
```

TABLE V-4

08/14/75 INPUT LISTING AUTOFLOW CHART SET - FINAL -TPFMOD- FLOWCHARTS

LISTING - PROGRAM TPFMOD

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
C TPF INPUT CREATE/MODIFY PROGRAM 13JUN75
C SUBROUTINES CALLED-MSG, CONVRT, DECODE, COMND
      IMPLICIT INTEGER*2 ($)
      INTEGER*4 $1,$16,$35,$36
      REAL*4 $5,$11,$12,$13,$14,$18,$25,$31,$32
      REAL*8 $2A,$3A,$3B
      INTEGER*2 YES/'Y'/, NO/'N'/
      COMMON /A/ $1(144),$2A(144),$3A(144),$4(144),$5(144),$6(144)
      COMMON /A/ $7(144),$8(144),$9(144),$10(144),$11(144),$12(144)
      COMMON /A/ $13(144),$14(144),$15(144),$16(144),$17(144),$18(144)
      COMMON /A/ $19(144),$20(144),$21(144),$22(144),$23(144),$24(144)
      COMMON /A/ $25(144),$26(144),$27(144),$28(144),$29(144),$30(144)
      COMMON /A/ $31(144),$32(144),$33(144),$34(144),$35(144),$36(144)
      COMMON /A/ $2B (144),$3B(144)
      INTEGER*2 IN(5) /5*0/, IN2 (5) /5*0/, CODE, NJ/6/,JN
      COMMON NGR
      INTEGER*4 NEW /'NEW '/, QUIT /'QUIT'/, DONE /'DONE'/, TEXT, TCDP
      DEFINE FILE 3 (500,80,E,J3)
      EXTERNAL RANGE
      CALL ERRSET (232,100,-1,1,RANGE)
      CALL COMND (NJ,JN)
      N = 0
      NGR = 0
 1000 WRITE (6,1)
    1 FORMAT (/' TPF COURSE CREATE/MODIFY PROGRAM'/)
    2 WRITE (6,3)
    3 FORMAT (' +ENTER ''NEW'' OR ''COPY'' '/)
    4 READ (5,5) IN
    5 FORMAT (10A1)
      CALL MSG (IN, CODE)
      IF (CODE .LE. 2) GO TO 999
      IF (CODE .EQ. 6) GO TO 100
      IF (CODE .EQ. 7) GO TO 200
      IF (CODE .EQ. 8) GO TO 300
      IF (CODE .EQ. 9) GO TO 400
      WRITE (6,6) IN
    6 FORMAT (1X, 5A1, ' +INVALID- RETRY'/)
      GO TO 2
C ***** COPY ***** COPY EXISTING COURSE FROM FILE 3
  100 WRITE (6,101)
  101 FORMAT (/ ' +COPY EXISTING COURSE- ENTER CDP')
  102 READ (5,103) TEXT
  103 FORMAT (A4)
      IF (TEXT .EQ. QUIT) GO TO 999
      IF (TEXT .EQ. DONE) GO TO 1000
      IF (TEXT .EQ. NEW)  GO TO 200
      IF (TEXT .EQ. ALL)  GO TO 300
      IF (TEXT .EQ. HELP) CALL HELP1
      IF (TEXT .EQ. HELP) GO TO 100
C HERE IF DIDNT RECOGNIZE MESSAGE- SEE IF IS IN CDP FILE
      DO 110 I = 1, 400, 2
      READ (3'I, 103, ERR=115) TCDP
      IF (NGR .GT. 0) GO TO 115
      IF (TCDP .EQ. TEXT) GO TO 120
  110 CONTINUE
C HERE IF ERROR OR NOT IN FILE
  115 NGR = 0
      WRITE (6,116) TEXT
  116 FORMAT (1X, A4, ' +NOT IN FILE- RETRY OR ENTER DONE'/)
      GO TO 102
C HERE IF FOUND CDP
  120 CONTINUE
      N = N + 1
      J = I
      READ (3'J,188,ERR=115) $1(N),$2A(N),$2B(N),$3A(N),$3B(N),$4(N),$5(
     *N), $6(N),$7(N),$8(N),$9(N),$10(N),$11(N),$12(N),$13(N),$14(N),
     * $15(N),$16(N),$17(N),$18(N),$19(N),$20(N),$21(N),$22(N),$23(N),
     * $24(N),$25(N),$26(N),$27(N),$28(N),$29(N),$30(N),$31(N),
     * $32(N),$33(N),$34(N),$35(N),$36(N)
  188 FORMAT (A4,A8,A2,2A8,I3,F3.1,2I3,2I5,I2,4F4.3,I4,A4,2X,/,4X,
     * 2(I3,F3.1,2I3,2I5,I3),2F4.1,2A2,2A4)
  121 WRITE (6,122) TEXT
  122 FORMAT (/ ' CDP ', A4, ' +COPIED- ENTER ''DONE'' OR ''MOD'' ')
  123 READ (5,5) IN,IN2
      CALL MSG (IN, CODE)
      GO TO (999, 1000, 124, 126, 125, 100, 200, 300, 400), CODE
      WRITE (6,6) IN
      GO TO 123
  124 CALL HELP1
      GO TO 121
  125 CALL SHOW (N)
      GO TO 121
  126 I = 9999
      CALL DECODE (I,N,CODE,IN,IN2)
      IF (CODE .EQ. 0) GO TO 121
      WRITE (6,6) IN2
      GO TO 123
C **** NEW **** CREATE NEW COURSE- PROMPT FOR ENTRIES
  200 CALL COMND (NJ,JN)
  201 WRITE (6,202)
  202 FORMAT (/' CREATE NEW COURSE'/)
      N1 = N + 1
      I = 1
      CALL DECODE (I,N1,CODE,IN,IN2)
      IF (CODE .EQ. 0) GO TO 210
      GO TO (999,1000,204,203,203,100,200,300), CODE
  203 WRITE (6,7)
    7 FORMAT (' +INVALID- RETRY'/)
      GO TO 201
  204 CALL HELP1
      GO TO 201
C HERE IF FOUND CDP- NOW INCRIMENT COUNTER & CREATE COURSE
  210 WRITE (6,211)
  211 FORMAT (/' ENTER FOLLOWING DATA TO CREATE COURSE'/)
      N = N + 1
      DO 230 I = 3, 16
      IF (I .EQ.10) GO TO 230
      IF (I .EQ.14) GO TO 230
  212 II = I
```

TABLE V-5

```
CARD NO   ****                    CONTENTS                                  ****

110              IF (I .EQ. 16) I = 36                                       00011000
111              CALL DECODE (I,N,CODE,IN,IN2)                               00011100
112              IF (CODE .EQ. 0) GO TO 230                                  00011200
113              GO TO (999,1000,221,212,222,100,200,300,400), CODE          00011300
114              WRITE (6,6) IN                                              00011400
115              GO TO 212                                                   00011500
116          221 CALL HELP1                                                  00011600
117              GO TO 212                                                   00011700
118          222 CALL SHOW (N)                                               00011800
119              GO TO 212                                                   00011900
120          230 WRITE (6,829)                                               00012000
121          229 FORMAT (' ')                                                00012100
122        C END PROMPTED COURSE CREATE                                      00012200
123          231 WRITE (6,232)                                               00012300
124          232 FORMAT (/' KEY DATA ENTERED FOR CREATE- ENTER ''MOD'' OR ''DONE'' 00012400
125             ')                                                           00012500
126          236 READ (5,5) IN,IN2                                           00012600
127              CALL MSG (IN,CODE)                                          00012700
128              GO TO (999,1000,233,235,234,100,200,300,400), CODE          00012800
129              WRITE (6,6) IN                                              00012900
130              GO TO 236                                                   00013000
131          233 CALL HELP1                                                  00013100
132              GO TO 231                                                   00013200
133          235 I = 9999                                                    00013300
134              CALL DECODE (I,N,CODE,IN,IN2)                               00013400
135              IF (CODE .EQ. 0) GO TO 231                                  00013500
136              WRITE (6,6) IN2                                             00013600
137              GO TO 236                                                   00013700
138          234 CALL SHOW (N)                                               00013800
139              GO TO 231                                                   00013900
140        C **** ALL **** HERE IF COPY ALL                                  00014000
141          300 WRITE (6,301)                                               00014100
142          301 FORMAT (/' ''ALL'' OPTION NOT SUPPORTED THIS PROGRAM' /)    00014200
143              GO TO 1000                                                  00014300
144        C **** MOD **** HERE IF MOD CALLED                                00014400
145          400 IF (N .EQ. 0) GO TO 412                                     00014500
146              WRITE (6,401) B1 (N)                                        00014600
147          401 FORMAT (/' MODIFY DATA FOR CDP ',A4)                        00014700
148          402 WRITE (6,403)                                               00014800
149          403 FORMAT (/' +ENTER FIELD TO BE CHANGED')                     00014900
150          404 READ (5,5) IN,IN2                                           00015000
151              I = 0                                                       00015100
152              CALL DECODE (I,N,CODE,IN,IN2)                               00015200
153              IF (CODE .EQ. 0) GO TO 402                                  00015300
154              GO TO (999,1000,410,402,411,100,200,300,400), CODE          00015400
155              WRITE (6,6) IN                                              00015500
156              GO TO 402                                                   00015600
157          410 CALL HELP1                                                  00015700
158              GO TO 402                                                   00015800
159          411 CALL SHOW (N)                                               00015900
160              GO TO 402                                                   00016000
161          412 WRITE (6,413)                                               00016100
162          413 FORMAT (' MOD INVALID- NO COURSE IN FILE'/)                 00016200
163              GO TO 1000                                                  00016300
164        C **** WRAPUP ****                                                00016400
165          999 WRITE (6,501)                                               00016500
166          501 FORMAT (/' SAVE NEW DATA BASE? (Y OR N)')                   00016600
167              READ (5,5) IN                                               00016700
168              IF (IN (1) .EQ. YES) GO TO 510                              00016800
169              IF (IN (1) .EQ. NO) GO TO 1999                              00016900
170              CALL MSG (IN,CODE)                                          00017000
171              GO TO (1999,1999,520,520,520,100,200,300,400),CODE          00017100
172          520 WRITE (6,6) IN                                              00017200
173              GO TO 999                                                   00017300
174          510 NSAVE = N                                                   00017400
175              DO 515 N = 1, NSAVE                                         00017500
176          515 WRITE (4,516) B1(N),B2A(N),B2B(N),B3A(N),B3B(N),B4(N),B5(   00017600
177             *N), B6(N),B7(N),B8(N),B9(N),B10(N),B11(N),B12(N),B13(N),B14(N), 00017700
178             * B15(N),B16(N),B1(N),B17(N),B18(N),B19(N),B20(N),B21(N),B22(N),B23 00017800
179             * (N),B24(N),B25(N),B26(N),B27(N),B28(N),B29(N),B30(N),B31(N), 00017900
180             * B32(N),B33(N),B34(N),B35(N),B36(N)                          00018000
181          516 FORMAT (A4,A8,A2,2A8,I3,F3.0,2I3,2I5,I2,4F4.3,I4,A4,2X,/,A4, 00018100
182             * 2I13,F3.0,2I3,2I5,I3),2F4.1,2A2,2A4)                       00018200
183              STOP                                                        00018300
184         1999 STOP 999                                                    00018400
185              END                                                         00018500
186              SUBROUTINE HELP1                                            00018600
187              INTEGER*4 TEXT,QUIT/'QUIT'/,SKIP/'SKIP'/,FIEL/'FIEL'/,CODE/'CODE'/00018700
188              INTEGER*4 DONE /'DONE'/                                     00018800
189            1 WRITE (6,3)                                                 00018900
190            3 FORMAT (/' +HELP CALLED- ENTER FIELD, CODES, OR SKIP')      00019000
191              READ (5,5) TEXT                                             00019100
192            5 FORMAT (A4)                                                 00019200
193              IF (TEXT .EQ. QUIT) GO TO 999                               00019300
194              IF (TEXT .EQ. DONE) GO TO 999                               00019400
195              IF (TEXT .EQ. SKIP) GO TO 999                               00019500
196              IF (TEXT .EQ. FIEL) GO TO 100                               00019600
197              IF (TEXT .EQ. CODE) GO TO 200                               00019700
198              WRITE (6,10) TEXT                                           00019800
199           10 FORMAT (1X, A4, ' INVALID- RETRY')                          00019900
200              GO TO 1                                                     00020000
201          100 WRITE (6,101)                                               00020100
202          101 FORMAT (//' TPF FIELDS ARE:'/                               00020200
203             *' CDP    CONV1  LEN1   CAP1   BCAP1  DMD1   BDMD1  OFST'/   00020300
204             *' CTLG   CONV2  LEN2   CAP2   BCAP2  DMD2   BDMD2  CHWK1'/  00020400
205             *' CRS    CONV3  LEN3   CAP3   BCAP3  DMD3   BDMD3  CHWK2'/  00020500
206             *' SCH    PFAIL  NADIS  NOSHO  SETB   BKLG'/                 00020600
207             *' JNO    PNEC   PADB   APAIL  PRTY   CTYPE'//)               00020700
208              GO TO 999                                                   00020800
209          200 WRITE (6,201)                                               00020900
210          201 FORMAT (//' VALID CODES ARE:'/                              00021000
211             *' QUIT  DONE  HELP  STAT  SHOW  COPY  NEW  ALL  MOD'//)     00021100
212              GO TO 999                                                   00021200
213          999 RETURN                                                      00021300
214              END                                                         00021400
215              SUBROUTINE RANGE (KK,LL)                                    00021500
216              COMMON MGR                                                  00021600
217              MGR = 10                                                    00021700
218              KK = 0                                                      00021800
219              LL = 232                                                    00021900
220              RETURN                                                      00022000
221              END                                                         00022100
```

TABLE V-5 (CONTINUED)

LISTING - SUBROUTINE DECODE

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

```
CARD NO      ****                    CONTENTS                    ****

  1    C DECODE SUBROUTINE FOR TPF 'WHAT IF' 22MAY75                    00000100
  2          SUBROUTINE DECODE (I,J,RESP,IN,IN2)                        00000200
  3          IMPLICIT INTEGER*2 (B)                                     00000300
  4          INTEGER*4 $1,$16,$35,$36                                   00000400
  5          REAL*4 $5,$11,$12,$13,$14,$18,$25,$31,$32                  00000500
  6          REAL*8 $2A,$3A,$3B                                         00000600
  7    C                                                                00000700
  8          COMMON /A/ $1(144),$2A(144),$3A(144),$4(144),$5(144),$6(144)  00000800
  9          COMMON /A/ $7(144),$8(144),$9(144),$10(144),$11(144),$12(144)  00000900
 10          COMMON /A/ $13(144),$14(144),$15(144),$16(144),$17(144),$18(144)  00001000
 11          COMMON /A/ $19(144),$20(144),$21(144),$22(144),$23(144),$24(144)  00001100
 12          COMMON /A/ $25(144),$26(144),$27(144),$28(144),$29(144),$30(144)  00001200
 13          COMMON /A/ $31(144),$32(144),$33(144),$34(144),$35(144),$36(144)  00001300
 14          COMMON /A/ $2B (144),$3B(144)                              00001400
 15          INTEGER*2 NUM (5) /5907/, CODE, TYPE (36), DEC, INT, ALPHA  00001500
 16          DATA TYPE /3*'A', 7*'I', 4*'F', 'I', 'A', 15*'I','F', 4*'A'/  00001600
 17          DATA ALPHA /'A'/, INT /'I'/, DEC /'F'/                     00001700
 18          REAL*8 ALPHA1, ALPHA2                                      00001800
 19          INTEGER*4 ANS, A4 (2),QUIT/'QUIT'/,DONE/'DONE'/,HELP/'HELP'/,SHOW/00001900
 20         *'SHOW'/                                                    00002000
 21          INTEGER*2 IN (5),IN2(5),F (5,36), COP (5), CTLG (5), CRS (5)  00002100
 22          INTEGER*2 CONV1 (5), LEN1 (5), CAP1 (5), BCAP1 (5), DMD1 (5)  00002200
 23          INTEGER*2 BDMD1 (5), OFST (5), PFAIL (5), NADIS (5), NOSHO (5)  00002300
 24          INTEGER*2 SETB (5), BKLG (5), JMO (5), CONV2 (5), LEN2 (5)  00002400
 25          INTEGER*2 CAP2 (5), BCAP2 (5), DMD2 (5), BDMD2 (5), CHWK2 (5)  00002500
 26          INTEGER*2 CONV3 (5), LEN3 (5), CAP3 (5), BCAP3 (5), DMD3 (5)  00002600
 27          INTEGER*2 BDMD3 (5), CHWK3 (5), PAOB (5), AFAIL (5), PRTY (5)  00002700
 28          INTEGER*2 CTYPE (5), PNEC (5), SCH (5), BLANK /' '/        00002800
 29          INTEGER*2 RESP, SIZE (36), A2 (4), NA2 (4)                 00002900
 30          DATA SIZE /4, 10, 16, 3, 4, 2*3, 2*5, 2, 4*5, 2*4, 3, 4,   00003000
 31         * 2*3, 2*5, 2*3, 4, 2*3, 2*5, 3, 4, 5, 2*2, 2*4/            00003100
 32          EQUIVALENCE (A2,ALPHA1), (A4,ALPHA1), (NA2,ALPHA2)         00003200
 33          EQUIVALENCE (COP, F(1,1)), (CTLG, F(1,2)), (CRS, F(1,3))   00003300
 34          EQUIVALENCE (CONV1, F(1,4)), (LEN1, F(1,5)), (CAP1, F(1,6))  00003400
 35          EQUIVALENCE (BCAP1, F(1,7)), (DMD1, F(1,8)), (BDMD1, F(1,9))  00003500
 36          EQUIVALENCE (OFST, F(1,10)), (PFAIL, F(1,11)), (NADIS, F(1,12))  00003600
 37          EQUIVALENCE (NOSHO, F(1,13)), (SETB, F(1,14)), (BKLG, F(1,15))  00003700
 38          EQUIVALENCE (JMO, F(1,16)), (CONV2, F(1,17)), (LEN2, F(1,18))  00003800
 39          EQUIVALENCE (CAP2, F(1,19)), (BCAP2, F(1,20)), (DMD2, F(1,21))  00003900
 40          EQUIVALENCE (BDMD2, F(1,22)), (CHWK2, F(1,23)), (CONV3, F(1,24))  00004000
 41          EQUIVALENCE (LEN3, F(1,25)), (CAP3, F(1,26)), (BCAP3, F(1,27))  00004100
 42          EQUIVALENCE (DMD3, F(1,28)), (BDMD3, F(1,29)), (CHWK3, F(1,30))  00004200
 43          EQUIVALENCE (PAOB, F(1,31)), (AFAIL, F(1,32)), (PRTY, F(1,33))  00004300
 44          EQUIVALENCE (CTYPE, F(1,34)), (PNEC, F(1,35)), (SCH, F(1,36))  00004400
 45          DATA COP/'C','O','P',' ',' '/,CTLG/'C','T','L','G',' '/    00004500
 46          DATA CRS/'C','R','S',' ',' '/,CONV1/'C','O','N','V','1'/   00004600
 47          DATA LEN1/'L','E','N','1',' '/,CAP1/'C','A','P','1',' '/   00004700
 48          DATA BCAP1/'B','C','A','P','1'/,DMD1/'D','M','D','1',' '/  00004800
 49          DATA BDMD1/'B','D','M','D','1'/,OFST/'O','F','S','T',' '/  00004900
 50          DATA PFAIL/'P','F','A','I','L'/,NADIS/'N','A','D','I','S'/  00005000
 51          DATA NOSHO/'N','O','S','H','O'/,SETB/'S','E','T','B',' '/  00005100
 52          DATA SCH/'S','C','H',' ',' '/,BKLG/'B','K','L','G',' '/    00005200
 53          DATA JMO/'J','M','O',' ',' '/,CONV2/'C','O','N','V','2'/   00005300
 54          DATA LEN2/'L','E','N','2',' '/,CAP2/'C','A','P','2',' '/   00005400
 55          DATA BCAP2/'B','C','A','P','2'/,DMD2/'D','M','D','2',' '/  00005500
 56          DATA BDMD2/'B','D','M','D','2'/,CHWK2/'C','H','W','K','2'/  00005600
 57          DATA CONV3/'C','O','N','V','3'/,LEN3/'L','E','N','3',' '/   00005700
 58          DATA CAP3/'C','A','P','3',' '/,BCAP3/'B','C','A','P','3'/  00005800
 59          DATA DMD3/'D','M','D','3',' '/,BDMD3/'B','D','M','D','3'/  00005900
 60          DATA CHWK3/'C','H','W','K','3'/,PAOB/'P','A','O','B',' '/  00006000
 61          DATA AFAIL/'A','F','A','I','L'/,PRTY/'P','R','T','Y',' '/  00006100
 62          DATA CTYPE/'C','T','Y','P','E'/,PNEC/'P','N','E','C',' '/  00006200
 63          RESP = 0                                                   00006300
 64          ISAVE = I                                                  00006400
 65          NEWFL = 1                                                  00006500
 66          IF (I .EQ. 9999) I = 0                                     00006600
 67    C IF I HAS VALUE ALREADY KNOW CODE- SKIP                         00006700
 68          IF (I .GT. 0) GO TO 100                                    00006800
 69    C DONT KNOW IF FIELD OR MSG- SSE IF MSG                          00006900
 70          NEWFL = 0                                                  00007000
 71          CALL MSG (IN,CODE)                                         00007100
 72          IF (CODE .GE. 98) GO TO 905                                00007200
 73    C HERE IF MSG- SEE IF STAT                                       00007300
 74          IF (CODE .EQ. 4) GO TO 900                                 00007400
 75    C NOT STAT- RETURN FROM SUBR WITH MSG CODE                       00007500
 76          RESP = CODE                                                00007600
 77          GO TO 999                                                  00007700
 78    C **** STAT **** - LOAD IN2 INTO IN                              00007800
 79      900 DO 902 J = 1,5                                             00007900
 80      902 IN(J) = IN2(J)                                             00008000
 81          ISAVE = 9999                                               00008100
 82    C SEARCH OUT FIELD NUMBER                                        00008200
 83      905 DO 910 J = 1,36                                            00008300
 84          DO 911 K = 1,5                                             00008400
 85          IF (IN (K) .NE. F (K,J)) GO TO 910                         00008500
 86      911 CONTINUE                                                   00008600
 87          I = J                                                      00008700
 88          IF (ISAVE .EQ. 9999) GO TO 200                             00008800
 89          GO TO 990                                                  00008900
 90      910 CONTINUE                                                   00009000
 91          RESP = 99                                                  00009100
 92          GO TO 999                                                  00009200
 93    C FIELD VALID- TEST IF DATA IN SECOND WORD                       00009300
 94      950 DO 960 J = 1,5                                             00009400
 95          IF (IN2(J) .NE. BLANK) GO TO 970                           00009500
 96      960 CONTINUE                                                   00009600
 97    C HERE IF BLANK FIELD- GO AND READ INPUT DATA                    00009700
 98          GO TO 100                                                  00009800
 99    C HERE IF SECOND DATA FIELD- BLANK DELIMIT                       00009900
100      970 K = 0                                                      00010000
101          DO 980 J = 1,5                                             00010100
102          NUM (J) = BLANK                                            00010200
103          IF (IN2(J) .EQ. BLANK) GO TO 980                           00010300
104          K = K + 1                                                  00010400
105          NUM (K) = IN2 (J)                                          00010500
106      980 CONTINUE                                                   00010600
107          GO TO 985                                                  00010700
108    C **** PROMPT DATA ENTRY ****                                    00010800
109      100 WRITE (6,101) (F (J,I),J = 1,5), TYPE (I), SIZE (I)        00010900
```

TABLE V-6

CARD NO **** CONTENTS ****

```
  101 FORMAT (' +ENTER ',5A1, 2X, '(',A1, I2, ')')
      IF (TYPE (I) .EQ. ALPHA) GO TO 110
      READ (5,102) NUM, IN2
  102 FORMAT (10A1)
  905 CALL CONVRT (NUM,ANS,CODE,DECMAL)
      IF (CODE .EQ.10) GO TO 105
      IF (CODE .EQ. 0) DECMAL = ANS
      GO TO 118
  105 CALL MSG (NUM, CODE)
      IF (CODE .LT. 98) GO TO 108
C HERE IF MSG DID NOT RECOGNIZE TEXT (CODE=99), PRINT ERROR
      IF (NEWFL .GT. 0) GO TO 1106
      DO 1107 J = 1,5
 1107 IN (J) = NUM (J)
      GO TO 905
 1106 WRITE (6,106) NUM
  106 FORMAT (1X, 5A1, ' +INVALID- RETRY'/)
      GO TO 100
  108 RESP = CODE
      IF (CODE .EQ. 4) GO TO 900
      GO TO 999
  110 READ (5,111) ALPHA1, ALPHA2
  111 FORMAT (2A8)
      RESP = 0
      IF (A4 (1) .EQ. QUIT) RESP = 1
      IF (A4 (1) .EQ. DONE) RESP = 2
      IF (A4 (1) .EQ. HELP) RESP = 3
      IF (A4 (1) .EQ. SHOW) RESP = 5
      IF (RESP .NE. 0) GO TO 999
      IF (SIZE (I) .EQ. 4) GO TO 112
      IF (SIZE (I) .EQ. 2) GO TO 113
      IF (SIZE (I) .EQ. 10) GO TO 114
      IF (SIZE (I) .EQ. 16) WRITE (6,1111) ALPHA1, ALPHA2
 1111 FORMAT (1X,2A8)
      GO TO 121
  112 WRITE (6,115) A4 (1)
  115 FORMAT (1X,A4)
      GO TO 121
  113 WRITE (6,116) A2 (1)
  116 FORMAT (1X,A2)
      GO TO 121
  114 WRITE (6,117) ALPHA1,NA2 (1)
  117 FORMAT (1X, A8, A2)
      GO TO 121
  118 IF (TYPE (I) .EQ. DEC) GO TO 1118
C HERE IF INTEGER
      ITEST = 10 ** SIZE (I)
      IF (ANS .GE. ITEST) GO TO 1119
      IF (ANS .GT. 32760) GO TO 1119
      WRITE (6,119) ANS
      GO TO 121
 1119 WRITE (6,1120) ANS
 1120 FORMAT (1X,I5,' TOO LARGE- RETRY'/)
      GO TO 100
 1118 WRITE (6,120) DECMAL
  119 FORMAT (1X, I5)
  120 FORMAT (F9.3)
  121 CONTINUE
C **** WRITE TO CRS ****
      IF (I .GT. 10) GO TO 710
      GO TO (1,2,3,4,5,6,7,8,9,10),I
    1 $1(N) = A4(1)
      GO TO 999
    2 $2A(N) = ALPHA1
      $2B(N) = NA2 (1)
      GO TO 999
    3 $3A(N) = ALPHA1
      $3B(N) = ALPHA2
      GO TO 999
    4 $4 (N) = ANS
      GO TO 999
    5 $5 (N) = ANS
      GO TO 999
    6 $6 (N) = ANS
      GO TO 999
    7 $7(N) = ANS
      GO TO 999
    8 $8 (N) = ANS
      GO TO 999
    9 $9(N) = ANS
      GO TO 999
   10 $10(N) = ANS
      GO TO 999
  710 IF (I .GT. 20) GO TO 720
      II = I - 10
      GO TO (11,12,13,14,15,16,17,18,19,20),II
   11 $11(N) = DECMAL
      GO TO 999
   12 $12(N) = DECMAL
      GO TO 999
   13 $13(N) = DECMAL
      GO TO 999
   14 $14(N) = DECMAL
      GO TO 999
   15 $15 (N) = ANS
      GO TO 999
   16 $16(N) = A4(1)
      GO TO 999
   17 $17 (N) = ANS
      GO TO 999
   18 $18(N) = ANS
      GO TO 999
   19 $19(N) = ANS
      GO TO 999
   20 $20(N) = ANS
      GO TO 999
  720 II = I - 20
      GO TO (21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36),II
   21 $21(N) = ANS
      GO TO 999
   22 $22(N) = ANS
      GO TO 999
```

TABLE V-6 (CONTINUED)

```
CARD NO      ****                    CONTENTS                                    ****

222            23 $23(N) = ANS                                                   00022200
223               GO TO 999                                                      00022300
224            24 $24 (N) = ANS                                                  00022400
225               GO TO 999                                                      00022500
226            25 $25(N) = ANS                                                   00022600
227               GO TO 999                                                      00022700
228            26 $26(N) = ANS                                                   00022800
229               GO TO 999                                                      00022900
230            27 $27(N) = ANS                                                   00023000
231               GO TO 999                                                      00023100
232            28 $28(N) = ANS                                                   00023200
233               GO TO 999                                                      00023300
234            29 $29(N) = ANS                                                   00023400
235               GO TO 999                                                      00023500
236            30 $30(N) = ANS                                                   00023600
237               GO TO 999                                                      00023700
238            31 $31(N) = ANS                                                   00023800
239               GO TO 999                                                      00023900
240            32 $32 (N) = DECMAL                                               00024000
241               GO TO 999                                                      00024100
242            33 $33(N) = ANS                                                   00024200
243               GO TO 999                                                      00024300
244            34 $34(N) = A2(1)                                                 00024400
245               GO TO 999                                                      00024500
246            35 $35(N) = A4(1)                                                 00024600
247               GO TO 999                                                      00024700
248            36 $36(N) = A4(1)                                                 00024800
249               GO TO 999                                                      00024900
250         C **** STAT **** SHOW (STAT EQUATIONS - HERE IF HAVE FOUND FIELD     00025000
251           200 IF (I .GT. 10) GO TO 210                                       00025100
252               GO TO (301,302,303,304,305,306,307,308,309,310),I               00025200
253           301 A4(1) = $1(N)                                                  00025300
254               GO TO 404                                                      00025400
255           302 ALPHA1 = $2A(N)                                                00025500
256               NA2(1) = $2B (N)                                               00025600
257               GO TO 410                                                      00025700
258           303 ALPHA1 = $3A(N)                                                00025800
259               ALPHA2 = $3B(N)                                                00025900
260               GO TO 416                                                      00026000
261           304 ANS = $4 (N)                                                   00026100
262               GO TO 400                                                      00026200
263           305 DECMAL = $5(N)                                                 00026300
264               GO TO 401                                                      00026400
265           306 ANS = $6(N)                                                    00026500
266               GO TO 400                                                      00026600
267           307 ANS = $7(N)                                                    00026700
268               GO TO 400                                                      00026800
269           308 ANS = $8(N)                                                    00026900
270               GO TO 400                                                      00027000
271           309 ANS = $9(N)                                                    00027100
272               GO TO 400                                                      00027200
273           310 ANS = $10(N)                                                   00027300
274               GO TO 400                                                      00027400
275           210 IF (I .GT. 20) GO TO 220                                       00027500
276               II = I - 10                                                    00027600
277               GO TO (311,312,313,314,315,316,317,318,319,320),II             00027700
278           311 DECMAL = $11(N)                                                00027800
279               GO TO 401                                                      00027900
280           312 DECMAL = $12(N)                                                00028000
281               GO TO 401                                                      00028100
282           313 DECMAL = $13(N)                                                00028200
283               GO TO 401                                                      00028300
284           314 DECMAL = $14(N)                                                00028400
285               GO TO 401                                                      00028500
286           315 ANS = $15(N)                                                   00028600
287               GO TO 400                                                      00028700
288           316 A4(1) = $16(N)                                                 00028800
289               GO TO 404                                                      00028900
290           317 ANS = $17(N)                                                   00029000
291               GO TO 400                                                      00029100
292           318 DECMAL = $18(N)                                                00029200
293               GO TO 401                                                      00029300
294           319 ANS = $19(N)                                                   00029400
295               GO TO 400                                                      00029500
296           320 ANS = $20(N)                                                   00029600
297               GO TO 400                                                      00029700
298           220 II = I - 20                                                    00029800
299               GO TO (321,322,323,324,325,326,327,328,329,330,331,332,333,334,335000029900
300                 *,336),II                                                    00030000
301           321 ANS = $21(N)                                                   00030100
302               GO TO 400                                                      00030200
303           322 ANS = $22(N)                                                   00030300
304               GO TO 400                                                      00030400
305           323 ANS = $23(N)                                                   00030500
306               GO TO 400                                                      00030600
307           324 ANS = $24(N)                                                   00030700
308               GO TO 400                                                      00030800
309           325 DECMAL = $25(N)                                                00030900
310               GO TO 401                                                      00031000
311           326 ANS = $26(N)                                                   00031100
312               GO TO 400                                                      00031200
313           327 ANS = $27(N)                                                   00031300
314               GO TO 400                                                      00031400
315           328 ANS = $28(N)                                                   00031500
316               GO TO 400                                                      00031600
317           329 ANS = $29(N)                                                   00031700
318               GO TO 400                                                      00031800
319           330 ANS = $30(N)                                                   00031900
320               GO TO 400                                                      00032000
321           331 DECMAL = $31(N)                                                00032100
322               GO TO 401                                                      00032200
323           332 DECMAL = $32(N)                                                00032300
324               GO TO 401                                                      00032400
325           333 A2(1) = $33(N)                                                 00032500
326               GO TO 402                                                      00032600
327           334 A2(1) = $34(N)                                                 00032700
328               GO TO 402                                                      00032800
329           335 A4(1) = $35(N)                                                 00032900
330               GO TO 404                                                      00033000
331           336 A4(1) = $36(N)                                                 00033100
332               GO TO 404                                                      00033200
333           400 WRITE (6,500) (F(J,I),J = 1,5),ANS                             00033300
```

TABLE V-6 (CONTINUED)

08/14/75 INPUT LISTING AUTOFLOW CHART SET - FINAL -DECODE- FLOWCHARTS

CARD NO **** CONTENTS ****

```
  500 FORMAT (/2X,5A1,1X,I5/)                                    00033400
      GO TO 999                                                  00033500
  401 WRITE (6,501) (F (J,I),J = 1,5),DECMAL                     00033600
  501 FORMAT (/2X,5A1,F9.3/)                                     00033700
      GO TO 999                                                  00033800
  404 WRITE (6,504) (F (J,I),J = 1,5),A4(1)                      00033900
  504 FORMAT (/2X,5A1,2X,A4/)                                    00034000
      GO TO 999                                                  00034100
  402 WRITE (6,502) (F (J,I),J = 1,5),A2(1)                      00034200
  502 FORMAT (/2X,5A1,A2,/)                                      00034300
      GO TO 999                                                  00034400
  410 WRITE (6,510) (F (J,I),J = 1,5),ALPHA1,NA2(1)              00034500
  510 FORMAT (/2X,5A1,A8,A2/)                                    00034600
      GO TO 999                                                  00034700
  416 WRITE (6,516) (F (J,I),J = 1,5),ALPHA1,ALPHA2              00034800
  516 FORMAT (/2X,5A1,A8,A8/)                                    00034900
  999 RETURN                                                     00035000
      END                                                        00035100
```

TABLE V-6 (CONTINUED)

08/14/75 INPUT LISTING AUTOFLOW CHART SET - FINAL -CONVRT- FLOWCHARTS

LISTING - SUBROUTINE CONVRT

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

```
CARD NO      ****                    CONTENTS                                  ****

  1        C CONVERSION SUBROUTINE - UPDATE 20MAY75                            00000100
  2        C CONVERSION OF 5 DIGET CHARACTER TO NUM                            00000200
  3        C RETURN CODE 0 FOR VALID CONVERT, 10 FOR INVALID                   00000300
  4              SUBROUTINE CONVRT (IN, ANS, CODE, DECMAL)                     00000400
  5              INTEGER*2 IN (5) , WORK (10), CODE                            00000500
  6              INTEGER*4 UPLIM, LOLIM, BLANK, HOLD (5), TEST, OUT (5), ANS,POINT 00000600
  7              EQUIVALENCE (HOLD, WORK)                                      00000700
  8              DATA WORK/10*0/,TEST/ZF000/,UPLIM/ZF940/,LOLIM/ZF040/,BLANK/Z4040/00000800
  9              DATA POINT /Z4B40/                                            00000900
 10              ANS = 0                                                       00001000
 11              FLG = 0                                                       00001100
 12              FRACT = 0                                                     00001200
 13              J = 1                                                         00001300
 14              DO 33 I = 1,5                                                 00001400
 15              K = I + I                                                     00001500
 16           33 WORK (K) = IN (I)                                             00001600
 17              K = 1                                                         00001700
 18        C CONVERT LOOP                                                      00001800
 19           35 DO 100 I = K, 5                                               00001900
 20              IF (HOLD (I) .EQ. BLANK) GO TO 200                            00002000
 21              IF (HOLD (I) .NE. POINT) GO TO 40                             00002100
 22              IF (FLG .GT. 0) GO TO 300                                     00002200
 23              FLG = 1                                                       00002300
 24              GO TO 200                                                     00002400
 25           40 IF (HOLD (I) .GT. UPLIM) GO TO 300                            00002500
 26              IF (HOLD (I) .LT. LOLIM) GO TO 300                            00002600
 27        C HERE IF VALID NUM                                                 00002700
 28          100 OUT (I) = (HOLD (I) - TEST) / 256                             00002800
 29              GO TO 210                                                     00002900
 30        C ASSEMBLE ANSWER-SET K IN CASE OF DECIMAL AT I                     00003000
 31          200 K = I + 1                                                     00003100
 32          201 I = I - 1                                                     00003200
 33          210 IF (FLG .GT. 1) GO TO 260                                     00003300
 34              IF (I) 250, 250, 220                                          00003400
 35          220 ANS = ANS + (OUT (I) * J)                                     00003500
 36              J = J * 10                                                    00003600
 37              GO TO 201                                                     00003700
 38        C END ANSWER LOOP                                                   00003800
 39        C TEST IF REAL OR FRACT                                             00003900
 40          250 IF (FLG .LE. 0) GO TO 295                                     00004000
 41        C HERE IF HAVE DECODED REAL PORTION-NOW GET FRACTION                00004100
 42              IF (K .GT. 5) GO TO 275                                       00004200
 43              K1 = K                                                        00004300
 44              FLG = 2                                                       00004400
 45              GO TO 35                                                      00004500
 46        C FRACTION LOOP                                                     00004600
 47          260 A = 1                                                         00004700
 48              DO 270 L = K1, I                                              00004800
 49              IF (I1 - L) .LT. 0) GO TO 275                                 00004900
 50              A = A * .1                                                    00005000
 51          270 FRACT = FRACT + (OUT (L) * A)                                 00005100
 52          275 DECMAL = ANS + FRACT                                          00005200
 53              CODE = 5                                                      00005300
 54              RETURN                                                        00005400
 55        C VALID INTEGER -RETURN                                             00005500
 56          295 CODE = 0                                                      00005600
 57              RETURN                                                        00005700
 58        C ALPHA - SET ERROR                                                 00005800
 59          300 CODE = 10                                                     00005900
 60              RETURN                                                        00006000
 61              END                                                           00006100
```

TABLE V-7 (CONTINUED)

LISTING - SUBROUTINE MSG

FORTRA[illegible] MODULE (LIST,NAMSG,SOURCESEQ)

```
CARD NO      ****                    CONTENTS                                    ****

   1       C SUBROUTINE TO DECODE TEXT MSG FOR 'WHAT IF' (TPF) 13JUN75            00000100
   2       C CODE- QUIT (1), DONE (2), HELP (3), STAT (4), SHOW (5), COPY (6),    00000200
   3       C       NEW (7), ALL (8), MOD (9), CODE NOT RECOGNIZED (99)            00000300
   4             SUBROUTINE MSG (IN, CODE)                                        00000400
   5             INTEGER*2 F(4,9), IN(5), CODE, QUIT(4), DONE(4), HELP(4), STAT(4) 00000500
   6             INTEGER*2 SHOW (4), COPY (4), NEW (4), ALL (4), MOD (4)          00000600
   7             EQUIVALENCE (QUIT,F(1,1)),(DONE,F(1,2)),(HELP,F(1,3)),(NEW,F(1,7))00000700
   8             EQUIVALENCE (SHOW,F(1,5)),(COPY,F(1,6)),(ALL,F(1,8)),(MOD,F(1,9)) 00000800
   9             EQUIVALENCE (STAT,F(1,4))                                        00000900
  10             DATA QUIT /'Q','U','I','T'/, DONE /'D','O','N','E'/              00001000
  11             DATA HELP /'H','E','L','P'/, SHOW /'S','H','O','W'/              00001100
  12             DATA COPY /'C','O','P','Y'/, NEW  /'N','E','W',' '/              00001200
  13             DATA ALL  /'A','L','L',' '/, MOD  /'M','O','D',' '/              00001300
  14             DATA STAT /'S','T','A','T'/                                      00001400
  15             CODE = 99                                                        00001500
  16             DO 2 I = 1,9                                                     00001600
  17             DO 1 J = 1,4                                                     00001700
  18             IF (IN (J) .NE. F(J,I)) GO TO 2                                  00001800
  19           1 CONTINUE                                                         00001900
  20       C HERE IF VALID COMPARE ON ALL 5 CHARACTERS                            00002000
  21             CODE = I                                                         00002100
  22             GO TO 99                                                         00002200
  23           2 CONTINUE                                                         00002300
  24          99 RETURN                                                           00002400
  25             END                                                              00002500
```

TABLE V-8

LISTING - SUBROUTINE SHOW

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

```
CARD NO      ****                    CONTENTS                    ****

      1    C DISPLAY CURRENT COURSE (TPFMOD), N IS CRS NO IN FILE 13JUN75
      2          SUBROUTINE SHOW (N)
      3          IMPLICIT INTEGER*2 (*)
      4          INTEGER*4 $1,$16,$35,$36
      5          REAL*4 $5,$11,$12,$13,$14,$18,$25,$31,$32
      6          REAL*8 $2A,$3A,$3B
      7          COMMON /A/ $1(144),$2A(144),$3A(144),$4(144),$5(144),$6(144)
      8          COMMON /A/ $7(144),$8(144),$9(144),$10(144),$11(144),$12(144)
      9          COMMON /A/ $13(144),$14(144),$15(144),$16(144),$17(144),$18(144)
     10          COMMON /A/ $19(144),$20(144),$21(144),$22(144),$23(144),$24(144)
     11          COMMON /A/ $25(144),$26(144),$27(144),$28(144),$29(144),$30(144)
     12          COMMON /A/ $31(144),$32(144),$33(144),$34(144),$35(144),$36(144)
     13          COMMON /A/ $2B (144),$3B(144)
     14          INTEGER*2 IN (5) /5*0/, BLANK /' '/, CODE
     15          I = N
     16          IF (I .LE. 0) GO TO 998
     17          IF (I .GT. 144) GO TO 998
     18        1 WRITE (6,2)
     19        2 FORMAT (/' SHOW: ENTER ALL OR FIELD NAME')
     20          READ (5,3) IN
     21        3 FORMAT (5A1)
     22          CALL MSG (IN, CODE)
     23          GO TO (999,999,10,10,10,10,10,100,10), CODE
     24    C HERE IF NOT RECOGNIZED-TEST IF BLANK(A RETURN)
     25          IF (IN (1) .EQ. BLANK) GO TO 100
     26    C NOT RECOGNIZED MSG OR BLANK-TEST IF FIELD NAME
     27          NN = 9999
     28          CALL DECODE (NN,N,CODE,IN)
     29          IF (CODE .EQ. 0) GO TO 999
     30       10 WRITE (6,11) IN
     31       11 FORMAT (1X,5A1,' *INVALID- RETRY'/)
     32          GO TO 1
     33    C ****SHOW ALL ****
     34      100 I = N
     35    C CALC STU BKLG- $4 CONV; $6 CAP; $7 BCAP; $15 BKLG WKS
     36          NBKL = 0
     37          IF ($4(I) .LE. 0) GO TO 299
     38          ICAP = $6(I) - $7(I)
     39          IF (ICAP .LE. 0) GO TO 299
     40          CNVB = 50. /$4(I)
     41          NBKL = $15(I)/CNVB * ICAP
     42      299 CONTINUE
     43    C **** PRINT IT ****
     44          WRITE (6,4001) $1(I),$3A(I),$3B(I)
     45          WRITE (6,4002) $4(I),$17(I),$24(I),NBKL
     46          WRITE (6,4003) $5(I),$18(I),$25(I),$15(I)
     47          WRITE (6,4004) $3A(I),$16(I),$2A(I),$28(I),$35(I)
     48         *, $6(I),$19(I),$26(I)
     49          WRITE (6,4005) $7(I),$20(I),$27(I),$23(I)
     50          WRITE (6,4006) $8(I),$21(I),$28(I),$30(I)
     51          WRITE (6,4007) $31(I),$32(I),$34(I),$33(I),$9(I),$22(I),$29(I)
     52         *, $10(I)
     53          RATEA = $11(I) * 100.
     54          RATEB = $12(I) * 100.
     55          RATEC = $13(I) * 100.
     56          WRITE (6,4008) RATEA, RATEB, RATEC
     57   4001 FORMAT (//1X,A4, 5X, 2A8, 4X, '--SCHEDULE DATA--   FY   FY+1   FY+
     58         *2   CRS BACKLOG')
     59   4002 FORMAT (' ', T31, 'ANNUAL CONVENINGS', I5, 2I7, 4X, '# STU', I5)
     60   4003 FORMAT (' SCHL JNO  CATALOG NO  NEC    COURSE LENGTH DYS', F5.1,
     61         * 2F7.1, 4X, '# WKS', I5)
     62   4004 FORMAT (' ', 2 (A4, 1X), A8, A2, 1X, A4, 4X, 'COURSE CAPACITY',
     63         * 3I7)
     64   4005 FORMAT (' ', T31, 'BUPERS CAPACITY', 3I7, 3X, 'FY+1 WK', I4)
     65   4006 FORMAT (' P AOB  PFAIL  TYPE   PRTY    ANNUAL DEMAND', 2X, 3I7,
     66         * 3X, 'FY+2 WK', I4)
     67   4007 FORMAT (' ', F5.1, F6.1, '-', 3X,A2,5X,A2,5X, 'BUPERS DEMAND',
     68         * 2X, 3I7, 3X, 'OFFSET', I5)
     69   4008 FORMAT (/' RATES: FAIL',F5.1,'%  NADIS',F5.1,'%  NOSHOW',F5.1,'%
     70         *'///)
     71          GO TO 999
     72    998 WRITE (6,98) I
     73     98 FORMAT (/1X,I3,' COURSE NOT IN PROCESS- SHOW SKIPPED'/)
     74    999 RETURN
     75          END
     76          BLOCK DATA
     77          IMPLICIT INTEGER*2 (*)
     78          INTEGER*4 $1,$16,$35,$36
     79          REAL*4 $5,$11,$12,$13,$14,$18,$25,$31,$32
     80          REAL*8 $2A,$3A,$3B
     81          COMMON /A/ $1(144),$2A(144),$3A(144),$4(144),$5(144),$6(144)
     82          COMMON /A/ $7(144),$8(144),$9(144),$10(144),$11(144),$12(144)
     83          COMMON /A/ $13(144),$14(144),$15(144),$16(144),$17(144),$18(144)
     84          COMMON /A/ $19(144),$20(144),$21(144),$22(144),$23(144),$24(144)
     85          COMMON /A/ $25(144),$26(144),$27(144),$28(144),$29(144),$30(144)
     86          COMMON /A/ $31(144),$32(144),$33(144),$34(144),$35(144),$36(144)
     87          COMMON /A/ $2B (144),$3B(144)
     88          DATA $1,$16,$35,$36/5760* ' '/,$2A,$3A,$3B/4320* ' '/,$2B/' '/
     89          DATA $5,$11,$12,$13,$14,$18,$25,$31,$32 /129600./
     90          DATA $4,$6,$7,$8,$9,$10,$15,$17,$19,$20,$21,$22,$23,$24,$26,
     91         1 $27, $28, $29, $30,  $34 /2880 * 0/, $33 /' '/
     92          END
```

TABLE V-9

SUBROUTINE COMND LISTING

```
COMND    START
         STM     14,12,12(13)
         BALR    12,Ø
         USING   *,12
         LM      3,4,Ø(1)
         LH      5,Ø(3)
         C       5,=F'17'
         MVC     Ø(2,4),=X'ØØØ1'
         BC      2,END
         LA      7,TAB1
         AR      5,7
         MVC     MSG+1(1),Ø(5)
         LA      L,PLIST
         SVC     X'CA'
         DC      AL4(ERR)
         MVC     Ø(2,4),=X'ØØØØ'
         B       END
ERR      MVC     Ø(2,4),=X'ØØØ2'
END      MVI     12(13),X'FF'
         LM      14,12,12(13)
         SR      15,15
         BR      14
         DS      ØF
PLIST    DC      AL8'TYPE'
         DC      X'81'
         DC      AL3(MSG)
         DC      C'B'
         DC      AL3(2)
MSG      DC      X'7LØØ'
TAB1     DC      X'718848C9995939B878F9FFØ3ØAØCØFFAF3Ø5'
         END
```

TABLE V-10

LISTING - EXECUTIVE TPF

```
TPF EXEC

&TYPE OFF
-TESTIT &IF &INDEX EQ Ø &GOTO -NONE
&IF &1 EQ QUIT &GOTO -QUIT
&INDEX1 = 1
&IF &1 EQ MASTER &GOTO -GOTIT
&INDEX1 = 2
&IF &1 EQ SCRATCH &GOTO -GOTIT
&INDEX1 = 3
&IF &1 EQ VERIFY &GOTO -GOTIT
&INDEX1 = 4
&IF &1 EQ TEST &GOTO -TEST
&IF &1 EQ QUIT &EXIT
&IF &1 EQ COPY &PRINT TPF MODEL- COPY INVALID- ENTER QUIT TO EXIT
&IF &1 EQ MOD &PRINT TPF MODEL- MOD INVALID- ENTER QUIT TO EXIT
-NONE &PRINT REPLY MASTER, SCRATCH, OR TEST (INPUT DATA)
&READ ARGS
&GOTO -TESTIT
-TEST &PRINT COMBINE TEST FILE WITH MASTER OR SCRATCH? (Y OR N)
&INDEX2 = 1
&READ ARGS
&IF &1 NE Y &GOTO -GOTIT
&INDEX2 = 1Ø
&PRINT REPLY MASTER OR SCRATCH
&READ ARGS
&GOTO -TESTIT
-GOTIT &PRINT
&PRINT INPUT THIS RUN:
&IF &INDEX1 EQ 1 &PRINT MASTER
&IF &INDEX1 EQ 2 &PRINT SCRATCH
&IF &INDEX1 EQ 3 &PRINT VERIFY
&IF &INDEX1 EQ 4 &PRINT TEST
&IF &INDEX2 EQ 1Ø &PRINT COMBINED WITH TEST
&IF &INDEX LT 2 &GOTO -NOSAVE
&IF &2 NE SAVE &GOTO -NOSAVE
&INDEX3 = 1Ø
&PRINT SAVE OPTION IN EFFECT THIS RUN
-NOSAVE &PRINT ENTER CARRIAGE RETURN TO CONTINUE
&READ ARGS
&IF &INDEX EQ Ø &GOTO -PARM
&IF &1 EQ QUIT &GOTO -QUIT
-PARM FILEDEF * CLEAR
FILEDEF PARM DSK TPF PARM
PARAM
&IF &INDEXØ GE 2 &GOTO -QUIT
FILEDEF * CLEAR
FILEDEF 5 DSK TPF PARM
FILEDEF 6 DUM
FILEDEF 8 DUM
&IF &INDEX1 EQ 1 FILEDEF 1Ø DSK TPF MASTER
&IF &INDEX1 EQ 2 FILEDEF 1Ø DSK TPF SCRATCH
&IF &INDEX1 EQ 3 FILEDEF 1Ø DSK TPF VFY
&IF &INDEX1 EQ 4 FILEDEF 1Ø DSK TST DATA
&IF &INDEX2 EQ 1Ø FILEDEF 1Ø DSK TST DATA CONCAT
&IF &INDEX3 EQ 1Ø &GOTO -SAVEOUT
FILEDEF 9 DSK OUT4 FILE LR 81
FILEDEF 11 DSK OUT3 FILE LR 81
&GOTO -MODEL
-SAVEOUT FILEDEF 9 DSK OUT2 FILE LR 81
FILEDEF 11 DSK OUT1 FILE LR 81
-MODEL RUN KEND (BRIEF)
FILEDEF * CLEAR
SET SPIE ON
&PRINT END TPF EXECUTION
&PRINT ENTER: CTRRUN FOR CENTER SUMMARY; CRSRUN FOR COURSE OUTPUTS
&EXIT
-QUIT FILEDEF * CLEAR
SET SPIE ON
&PRINT MODEL NOT RUN AT OPERATOR REQUEST
&EXIT
```

TABLE V-11

LISTING - PROGRAM PARAM

PL/1 MODULE (LIST,NAMSQ,SOURCSEQ)

CARD NO **** CONTENTS ****

```
 1   OPTION PACKIF,COMPRESS
 2    PARAM: PROC OPTIONS (MAIN) ;                                          00000100
 3    /* TPF PARAMETER CARD MOD PROGRAM 13JUN75 */                          00000200
 4    DCL (PCARD,DATA) CHAR (80), REQ CHAR (10), LINE(8) C[illegible]10) VARYING, 00000300
 5       FIELDS (24) CHAR (5) INITIAL                                       00000400
 6      ('FAIL','NOSHO','DMD','TREND','CONV','LEN','CAP','SE[illegible]',   00000500
 7       'CDP','SCH','PRINT','WKS','TRAC','PRE','RELS','END','STAT',        00000600
 8       'CRS','SCHL','TR','INPUT','FYR','LYR','AOB'),                      00000700
 9       LEN [illegible] BIN (8,0) INITIAL                                  00000800
10       ((10,4,(10) [illegible]),                                          00000900
11       COL(24) FI[illegible]D BIN (8,0) INITIAL (16,21,26,31,36,41,       00001000
12       46,56,61,[illegible],71,72,73,74,75,76,77,78,79,80,1,3,7,11),      00001100
13       (K,LP,[illegible]1,P2) FIXED BIN (8,0) ;                           00001200
14    DCL COMND EXTERNAL ENTRY;                                             00001300
15    DCL IMESARC ENTRY (FIXED BIN (31,0));                                 00001400
16    DCL (II,JJ) FIXED BIN (15,0);                                         00001500
17    OPEN FILE (PARM) RECORD UPDATE ;                                      00001600
18    READ FILE (PARM) INTO (PCARD) ;                                       00001700
19    II = 61                                                               00001800
20    CALL COMND (II,JJ);                                                   00001900
21    INTRO:                                                                00002000
22    DISPLAY ('TPF MODEL - WANT INSTRUCTIONS (Y OR N)') REPLY (DATA);      00002100
23    IF DATA = 'N' THEN GO TO PRERUN ;                                     00002200
24    HELP: DISPLAY (' ');                                                  00002300
25    DISPLAY ('ENTER: FNAME XXXX  (SET FNAME TO VALUE)');                  00002400
26    DISPLAY ('        FNAME DEACT (DEACT FNAME)');                        00002500
27    DISPLAY ('        SHOW        (SHOW RUN COND)');                      00002600
28    DISPLAY ('        RUN         (RUN MODEL)');                          00002700
29    DISPLAY (' ');                                                        00002800
30    DISPLAY ('FIELDS:  FYR   (NNNN)  LYR   (NNNN)  AOB   (NNNN)  CDP   (AA00002900
31   AA)  SCH   (AAAA)');                                                   00003000
32    DISPLAY ('          FAIL  (NNNN)  NOSHO (NNNN)  DMD   (NNNN)  TREND (NN00003100
33   NN)  CONV  (NNNN)');                                                   00003200
34    DISPLAY ('          LEN   (NNNN)  CAP   (NNNN)  SETB  (NNNN)');       00003300
35    DISPLAY (' ');                                                        00003400
36    DISPLAY ('REPORTS: CRS   (N)     SCHL  (N)     CTR (N)       PRINT (N)00003500
37   ');                                                                    00003600
38    PRERUN:                                                               00003700
39    FLAG = 2;                                                             00003800
40    GO TO TUBE_IT;                                                        00003900
41    COME_BACK: FLAG = 0;                                                  00004000
42    DISPLAY (' ');                                                        00004100
43    DISPLAY ('ENTER RUN, QUIT, OR FIELD TO BE CHANGED');                  00004200
44    PARMS: DISPLAY (' ');                                                 00004300
45    DISPLAY ('ENTER REQUEST:') REPLY (DATA) ;                             00004400
46    IF DATA = 'RUN' THEN GO TO FILE_IT ;                                  00004500
47    IF DATA = 'FLAG' THEN GO TO FLG;                                      00004600
48    IF DATA = 'HELP' THEN GO TO HELP;                                     00004700
49    IF DATA = 'SHOW' THEN GO TO TUBE_IT ;                                 00004800
50    IF DATA = 'QUIT' THEN GO TO QUIT;                                     00004900
51    P1 = INDEX (DATA,' ') ;                                               00005000
52    P2 = INDEX (SUBSTR (DATA,P1+1),' ') ;                                 00005100
53    REQ = SUBSTR (DATA,1,P1-1) ;                                          00005200
54    DO I = 1 TO 24 ;                                                      00005300
55    IF REQ = FIELDS (I) THEN GO TO GOT_IT ;                               00005400
56    END;                                                                  00005500
57    DISPLAY ('FIELDNAME NOT RECOGNIZED') ;                                00005600
58    GO TO PARMS ;                                                         00005700
59    GOT_IT:                                                               00005800
60    IF SUBSTR (DATA,P1+1,6) = 'DEACT ' THEN GO TO DEAC ;                  00005900
61    IF SUBSTR (DATA,1,P1-1) = 'SCH' THEN                                  00006000
62    IF P2 < 5 THEN P2 = 5;                                                00006100
63    IF P2-1 ¬= LEN(I) THEN GO TO ERR2 ;                                   00006200
64    IF I <= 10 THEN GO TO FLAG_AND_VALU ;                                 00006300
65    SUBSTR (PCARD,COL(I),LEN(I)) = SUBSTR (DATA,P1+1,LEN(I)) ;            00006400
66    GO TO PARMS ;                                                         00006500
67    TUBE_IT:                                                              00006600
68    DISPLAY (' ');                                                        00006700
69    DISPLAY ('PRESENT RUN CONDITIONS');                                   00006800
70    DO K = 1 TO 8 ;                                                       00006900
71    LINE(K) = '' ;                                                        00007000
72    END;                                                                  00007100
73    LP = 0 ;                                                              00007200
74    DO K = 1 TO 7 ;                                                       00007300
75    IF SUBSTR (PCARD,11+5*K,1) ¬= '0' THEN DO ;                           00007400
76    LINE (LP/3 + 1) = LINE (LP/3 + 1) || FIELDS(K) ||                     00007500
77     ' = ' || SUBSTR (PCARD,12+5*K,4) || '      ' ;                       00007600
78    LP = LP + 1 ;                                                         00007700
79    END ;                                                                 00007800
80    END ;                                                                 00007900
81    DO K = 8 TO 10 ;                                                      00008000
82    IF SUBSTR (PCARD,16+5*K,1) ¬= '0' THEN DO ;                           00008100
83    LINE (LP/3 + 1) = LINE (LP/3 + 1) || FIELDS(K) ||                     00008200
84     ' = ' || SUBSTR (PCARD,17+5*K,4) || '      ' ;                       00008300
85    LP = LP + 1 ;                                                         00008400
86    END ;                                                                 00008500
87    END ;                                                                 00008600
88    LP = ((LP+3)/3) * 3;                                                  00008700
89    LINE (LP / 3 + 1) = 'REPORTS: ';                                      00008800
90    FLAG1 = 0;                                                            00008900
91    DO K = 11,18,19,20;                                                   00009000
92    IF SUBSTR (PCARD,COL (K),1) ¬='0' THEN DO;                            00009100
93    FLAG1 = 10;                                                           00009200
94    LINE (LP/3 + 1) = LINE (LP/3 + 1) || ' ' || FIELDS (K);               00009300
95    END;                                                                  00009400
96    END;                                                                  00009500
97    IF FLAG1 ¬= 10 THEN                                                   00009600
98    LINE (LP/3 + 1) = LINE (LP/3 + 1) || '** NONE **';                    00009700
99    LP = LP + 3;                                                          00009800
100   LINE (LP/3 + 1) = 'INPUT = ' || SUBSTR (PCARD,1,2) || '      FYR    = ' 00009900
101    || SUBSTR (PCARD,3,4) || '      LYR    = ' || SUBSTR (PCARD,7,4)      00010000
102    || '      AOB    = ' || SUBSTR (PCARD,11,4) ;                         00010100
103   K = (LP/3) + 1 ;                                                      00010200
104   DO I = 1 TO K ;                                                       00010300
105   DISPLAY (LINE(I)) ;                                                   00010400
106   END ;                                                                 00010500
107   IF FLAG = 2 THEN GO TO COME_BACK;                                     00010600
108   GO TO PARMS ;                                                         00010700
109   FLAG_AND_VALU:                                                        00010800
```

TABLE V-12

```
08/14/75        INPUT LISTING                          AUTOFLOW CHART SET - FINAL    PL/1 -PARAM- FLOWCHARTS

CARD NO         ****                          CONTENTS                                    ****

                SUBSTR (PCARD,COL(I),LEN(I)+1) =                                          00010900
                '1' || SUBSTR (DATA,PI+1,LEN(I)) ;                                        00011000
                GO TO PARMS ;                                                             00011100
                DEAC:                                                                     00011200
                IF I > 20 THEN GO TO ERR1 ;                                               00011300
                SUBSTR (PCARD,COL(I),1) = '0' ;                                           00011400
                GO TO PARMS ;                                                             00011500
                ERR1:                                                                     00011600
                DISPLAY ('INVALID REQUEST') ;                                             00011700
                GO TO PARMS ;                                                             00011800
                ERR2:                                                                     00011900
                DISPLAY ('INVALID DATA FORMAT') ;                                         00012000
                GO TO PARMS ;                                                             00012100
               FLG: DISPLAY ('FLAGS - WKS TRAC PRE RELS END STAT YEAR PRINT INIT');       00012200
                GO TO PARMS;                                                              00012300
                QUIT: II = 8;                                                             00012400
                CALL IHESARC (II);                                                        00012500
                FILE_IT:                                                                  00012600
                REWRITE FILE (PARM) FROM (PCARD) ;                                        00012700
                CLOSE FILE (PARM) ;                                                       00012800
                END PARAM ;                                                               00012900
```

TABLE V-12 (CONTINUED)

DISPLAY SUPPORT PROGRAM

The objective of the display support program is to allow the user to effectively interface with the large amounts of data that the TPF model may produce. This support package is general purpose in nature, and is useful for displaying the contents of other data sets as well.

EXECUTIVE DISP. Executive DISP is used to control the execution of program DISPLAY. This short executive accepts a parameter passed from the user or calling executive and uses this parameter to define the data set to be displayed. FORTRAN data set 7 is used as the output data set and is termed CONO by NCSS convention to label it the console or terminal device. This output data set is used rather than the conventional data set 6 so that FORTRAN I/O error messages will not be displayed. SET SPIE ON is specified in this executive as the FORTRAN I/O error checking is disabled by internal subroutines, and this condition must be reset. This executive is usually called by executives CTRRUN, CRSRUN, CENTER, or COURSE. These executives are three-line executives that merely pass the proper file name to the DISP exec. CTRRUN and CRSRUN are used to display center and course data respectively for TPF runs where the SAVE option is not specified, while CENTER and COURSE are used to display those data if SAVE was specified.

PROGRAM DISPLAY. Program DISPLAY is a FORTRAN program used to display the contents of TPF output data sets. The program ascertains the format of the output data, determines page size, and presents these data along with options to the user. A standard TPF output data set is in the form:

CTL CHAR(1)	DATA (79)
1	First line of page
Blank	Data (variable number of lines)
/	Last line of data
Blank	Null line
1	First line of next page

Program DISPLAY first searches the first column of each record for a "/", to determine if a TPF data set is to be displayed, and if so, to ascertain the size of each page. Next the program skips through the file to determine the number of pages in the file. Finally, if the file contains 25 lines per page, it is assumed to be a course file, and the CDP options are enacted. If the file is not a TPF file, the program defaults to 20 lines per page.

The user has the option of specifying the page number to be displayed, or entering a carriage return to display the next (or first) page. The user may also enter LOOK, followed by a blank and a page number to display only the first line of a page. Finally, if the file is a course file, the user may specify the CDP number of the course to be displayed.

Direct access I/O is used by this program in locating and displaying data so as to allow the user to browse back and forth through the data. All terminal responses are read in the "A" format as in program TPFMOD1 to allow the user to intermix alpha and numeric data and to avoid left or right justification. Subroutine CONVRT, detailed in the section describing TPFMOD1 is used to convert these input data to a machine processable form. Subroutine COMND, also described in that section is used to clear the display screen prior to a page

display. In-line subroutine RANGE is used to intercept FORTRAN errors should the user specify a page outside the boundaries of the input data set, while in-line subroutine ALPHA intercepts any I/O conversion errors in order to provide smooth operation to the user.

LISTING - DISPLAY EXECUTIVES

COURSE EXEC

```
&TYPE OFF
EXEC DISP OUT2
&EXIT
```

CENTER EXEC

```
&TYPE OFF
EXEC DISP OUT1
&EXIT
```

CRSRUN EXEC

```
&TYPE OFF
EXEC DISP OUT4
&EXIT
```

CTRRUN EXEC

```
&TYPE OFF
EXEC DISP OUT3
&EXIT
```

DISP EXEC

```
&TYPE OFF
FILEDEF * CLEAR
FILEDEF 3 DSK &1 FILE
FILEDEF 7 CONO
DISPLAY
SET SPIE ON
FILEDEF * CLEAR
&EXIT
```

TABLE V-13

LISTING - PROGRAM DISPLAY

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

```
CARD NO      ****                    CONTENTS                    ****

   1       C TPF SUPPORT - UPDATE 22MAY75                                  00000100
   2       C DISPLAY- DISPLAYS OUTPUT FILE FOR 3 MODES (AUTO SELECT OF STD/NON-STD)00000200
   3       C STANDARD FILE CONTAINS 1 IN COL 1 AT START OF PAGE, / AFTER LAST LINE 00000300
   4       C 1. COURSE FILE (25 LINES/PAGE) SELECT BY PAGE OR CDP.         00000400
   5       C 2. SCHOOL/CENTER FILE - SELECT BY PAGE NO.                    00000500
   6       C 3. NON STANDARD FILE - SELECT BY PAGE NO; DEFAULT TO 20 LINES/PAGE 00000600
   7             INTEGER*2 IN(5), IN2(5), CODE, TEST(4,5), SIZE,NSTART,NNN,ENDIT 00000700
   8             DATA TEST/'0','U','I','T','D','O','N','E','L','O','O','K',  00000800
   9            1 'P','A','G','E','C','D','P',' '/                          00000900
  10             INTEGER*2 BLANK /' '/                                      00001000
  11             REAL*4 CDP, DONE, PAGE, QUIT, SELECT                       00001100
  12             REAL*8 LINE (10)                                           00001200
  13             INTEGER*4 NUM                                              00001300
  14             COMMON NGA,NGR                                             00001400
  15             EXTERNAL RANGE                                             00001500
  16             EXTERNAL ALPHA                                             00001600
  17             DEFINE FILE 3 (2500,81,E,J3)                               00001700
  18             DATA SLASH/'/'/,CDP/'CDP '/,DONE/'DONE'/,PAGE/'PAGE'/,QUIT/'QUIT'/00001800
  19             CALL ERRSET (232,100,-1,1,RANGE)                           00001900
  20             CALL ERRSET (215,100,-1,1,ALPHA)                           00002000
  21             NGA = 0                                                    00002100
  22             NGR = 0                                                    00002200
  23             NSTART = - 1                                               00002300
  24             LOOKFL = 0                                                 00002400
  25             NSET = 2                                                   00002500
  26             NNN = 6                                                    00002600
  27             CALL COMND (NNN,J3)                                        00002700
  28       C LOAD PAGE SIZE TO CALC POSITION                                00002800
  29             DO 20 I = 1, 26                                            00002900
  30             READ (3'I,15,ERR=999) CTLCAR                               00003000
  31          15 FORMAT (A1,80X)                                            00003100
  32             IF (CTLCAR .EQ. SLASH) GO TO 30                            00003200
  33          20 CONTINUE                                                   00003300
  34       C HERE IF NO SLASH IN 1 - INVALID FILE                           00003400
  35             WRITE (7,25)                                               00003500
  36          25 FORMAT (' NON STANDARD FILE TYPE - DEFAULT 20 LINES PER PAGE') 00003600
  37             SIZE = 20                                                  00003700
  38             NSET = 0                                                   00003800
  39             GO TO 26                                                   00003900
  40       C                                                                00004000
  41          30 SIZE = I + 1                                               00004100
  42             WRITE (7,35) SIZE                                          00004200
  43          35 FORMAT (' STANDARD FILE - ',I2,' LINES')                   00004300
  44       C DETERMINE NUMBER OF PAGES IN FILE                              00004400
  45          26 DO 10026 I = 1, 100                                        00004500
  46             N = I * SIZE + 1                                           00004600
  47             READ (3'N,15,ERR=999) CTLCAR                               00004700
  48             IF (NGR) 10026, 10026, 10027                               00004800
  49       10026 CONTINUE                                                   00004900
  50       10027 NGR = 0                                                    00005000
  51             WRITE (7,10028) I                                          00005100
  52       10028 FORMAT (1X, I3, ' PAGES IN FILE')                          00005200
  53             IF (SIZE .NE. 25) GO TO 27                                 00005300
  54             IF (I .EQ. 1) GO TO 27                                     00005400
  55             WRITE (7, 40)                                              00005500
  56          40 FORMAT (/' *COURSE FILE - SELECT BY ''CDP'' OR ''PAGE'' ') 00005600
  57             GO TO 241                                                  00005700
  58          27 WRITE (7,36)                                               00005800
  59          36 FORMAT (/' *ENTER PAGE NUMBER')                            00005900
  60         241 READ (5,5) IN,IN2                                          00006000
  61           5 FORMAT (10A1)                                              00006100
  62             INFLG = 0                                                  00006200
  63       C TEST IF IN IS NUMBER OR CODE                                   00006300
  64         242 CALL CONVRT (IN,NUM,CODE,DUM)                              00006400
  65             IF (CODE .LE. 5) GO TO 666                                 00006500
  66       C HAS ALPHAS- SEE IF VALID CODE                                  00006600
  67             IF (INFLG .GT. 0) GO TO 247                                00006700
  68             INFLG = 10                                                 00006800
  69             DO 244 I = 1,5                                             00006900
  70             DO 243 J = 1,4                                             00007000
  71             IF (IN (J) .NE. TEST(J,I)) GO TO 244                       00007100
  72         243 CONTINUE                                                   00007200
  73       C HERE IF RECOGNIZED CODE                                        00007300
  74             GO TO (999,999,245,246,8001,1                              00007400
  75             GO TO 247                                                  00007500
  76         244 CONTINUE                                                   00007600
  77         247 WRITE (6,6) IN                                             00007700
  78           6 FORMAT (1X,5A1,' *INVALID- RETRY'/)                        00007800
  79             GO TO 241                                                  00007900
  80       C HERE IF LOOK- NEXT LINE IF PAGE                                00008000
  81         245 LOOKFL = 10                                                00008100
  82       C NOW TEST IF IN2 IS BLANK OR HAS NUM                            00008200
  83         246 DO 248 I = 1,5                                             00008300
  84             IF (IN2(I) .NE. BLANK) GO TO 249                           00008400
  85         248 CONTINUE                                                   00008500
  86       C HERE IF BLANK- GO SET NUM                                      00008600
  87             GO TO 27                                                   00008700
  88       C HAVE SECOND ENTRY- LOAD INTO IN & DECODE                       00008800
  89         249 DO 250 I = 1,5                                             00008900
  90         250 IN (I) = IN2 (I)                                           00009000
  91             INFLG = 10                                                 00009100
  92             GO TO 242                                                  00009200
  93         666 IF (NUM .NE. 0) GO TO 1666                                 00009300
  94             NSTART = NSTART + SIZE                                     00009400
  95             NUM = NSTART / SIZE + 1                                    00009500
  96       C CALC NSTART POSITION OF PAGE                                   00009600
  97        1666 NSTART = (NUM - 1) * SIZE + 1                              00009700
  98         667 ENDIT = NSTART + (SIZE - NSET)                             00009800
  99             IF (LOOKFL .GT. 1) ENDIT = NSTART                          00009900
 100       C CLEAR PAGE THEN OUTPUT                                         00010000
 101             CALL COMND (NNN,J3)                                        00010100
 102             IF (LOOKFL .EQ. 0) WRITE (7,103) NUM                       00010200
 103         103 FORMAT (' PAGE', I4)                                       00010300
 104             LOOKFL = 0                                                 00010400
 105             DO 110 I = NSTART,ENDIT                                    00010500
 106             READ (3'I,115,ERR=300) CTLCAR,LINE                         00010600
 107         115 FORMAT (A1,9A8,A7)                                         00010700
 108             IF (NGR) 120,120,300                                       00010800
 109         120 IF (CTLCAR .EQ. SLASH) GO TO 130                           00010900
```

TABLE V-14

520

```
CARD NO      ****                    CONTENTS                                  ****

110            110 WRITE (7,116) LINE                                          00011000
111            116 FORMAT (1X,9A8,A7)                                          00011100
112          C                                                                 00011200
113            140 WRITE (7,39) NUM                                            00011300
114                GO TO 241                                                   00011400
115             39 FORMAT (' PAGE', I4,'  +PAUSING- ENTER NEXT PAGE NO.')      00011500
116          C ALPHA ERROR                                                     00011600
117            200 NGA = 0                                                     00011700
118                WRITE (6,6) IN                                              00011800
119                GO TO 241                                                   00011900
120          C RANGE ERROR                                                     00012000
121            300 NGR = 0                                                     00012100
122                WRITE (7,310) NUM                                           00012200
123            310 FORMAT (1X,I4,1X,'OUT OF RANGE - RETRY')                    00012300
124                GO TO 241                                                   00012400
125          C                                                                 00012500
126          C CDP SELECT ROUTINE                                              00012600
127            800 WRITE (7,802)                                               00012700
128            802 FORMAT (/' +ENTER CDP')                                     00012800
129            803 READ (5,42,ERR=800) SELECT, IN2                             00012900
130             42 FORMAT (A4,1X,5A1)                                          00013000
131                IF (SELECT .EQ. QUIT) GO TO 999                             00013100
132                IF (SELECT .EQ. DONE) GO TO 999                             00013200
133                IF (SELECT .EQ. PAGE) GO TO 246                             00013300
134          C SEARCH IF CDP IN FILE                                           00013400
135                DO 816 I = 1, 100                                           00013500
136                N = ((I - 1) * SIZE) + 1                                    00013600
137                READ (3'N,806,ERR=850) CDPA                                 00013700
138            806 FORMAT (1X, A4)                                             00013800
139                IF (NGR)810, 810, 850                                       00013900
140            810 IF (CDPA .EQ. SELECT) GO TO 820                             00014000
141            816 CONTINUE                                                    00014100
142          C                                                                 00014200
143          C HAVE CDP IN FILE - PRINT PAGE                                   00014300
144            820 CALL COMND (NNN,J1)                                         00014400
145                NUM = 1                                                     00014500
146                ENDIT = N + 25                                              00014600
147                DO 825 I = N, ENDIT                                         00014700
148                READ (3'I,115,ERR=850) CTLCAR,LINE                          00014800
149                IF (NGR) 822, 822, 850                                      00014900
150            822 IF (CTLCAR .EQ. SLASH) GO TO 830                            00015000
151            825 WRITE (7,116) LINE                                          00015100
152          C END OF PAGE                                                     00015200
153            830 WRITE (7,832) NUM                                           00015300
154            832 FORMAT (' PAGE',I4,'  +PAUSING- ENTER CDP, PAGE, OR QUIT')  00015400
155                GO TO 803                                                   00015500
156          C HERE IF BAD READ / NO SELECT / OUT OF RANGE                     00015600
157            850 NGR = 0                                                     00015700
158                WRITE (7, 852) SELECT                                       00015800
159            852 FORMAT (1X, A4, ' +NOT IN FILE OR INVALID - RETRY')         00015900
160                GO TO 803                                                   00016000
161          C WRAPUP (END)                                                    00016100
162            999 WRITE (7,998)                                               00016200
163            998 FORMAT (' +DISPLAY TPF ENDED')                              00016300
164                STOP                                                        00016400
165                END                                                         00016500
166                SUBROUTINE RANGE (KK,LL)                                    00016600
167                COMMON NGA,NGR                                              00016700
168                NGR = 10                                                    00016800
169                KK = 0                                                      00016900
170                LL = 232                                                    00017000
171                RETURN                                                      00017100
172                END                                                         00017200
173                SUBROUTINE ALPHA (KK,LL)                                    00017300
174                COMMON NGA,NGR                                              00017400
175                NGA = 10                                                    00017500
176                KK = 0                                                      00017600
177                LL = 215                                                    00017700
178                RETURN                                                      00017800
179                END                                                         00017900
```

TABLE V-14 (CONTINUED)

CHART TITLE - PROCEDURES

AUTOFLOW - PROGRAM DISPLAY

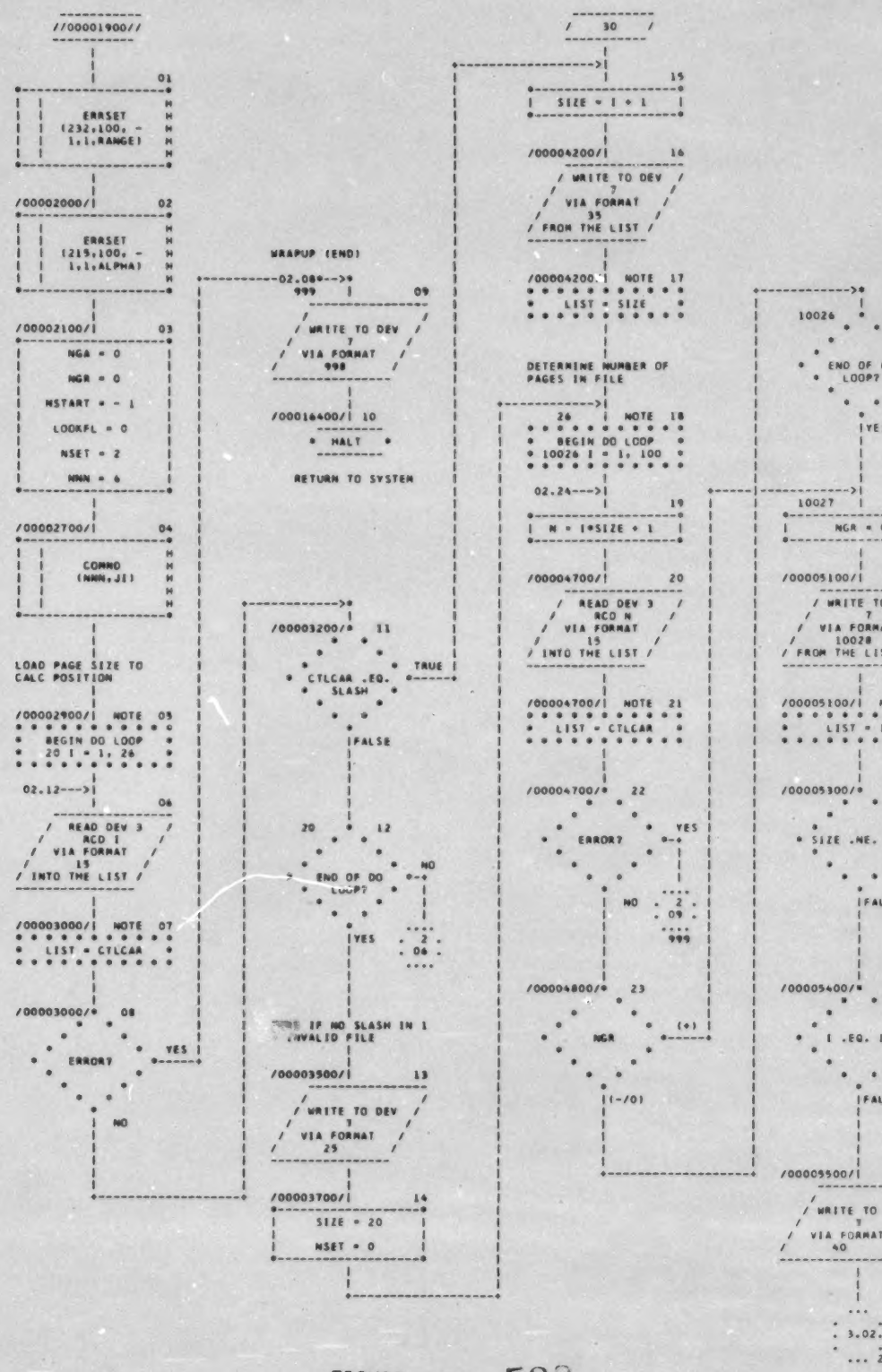


FIGURE V-14 522

CHART TITLE - PROCEDURES

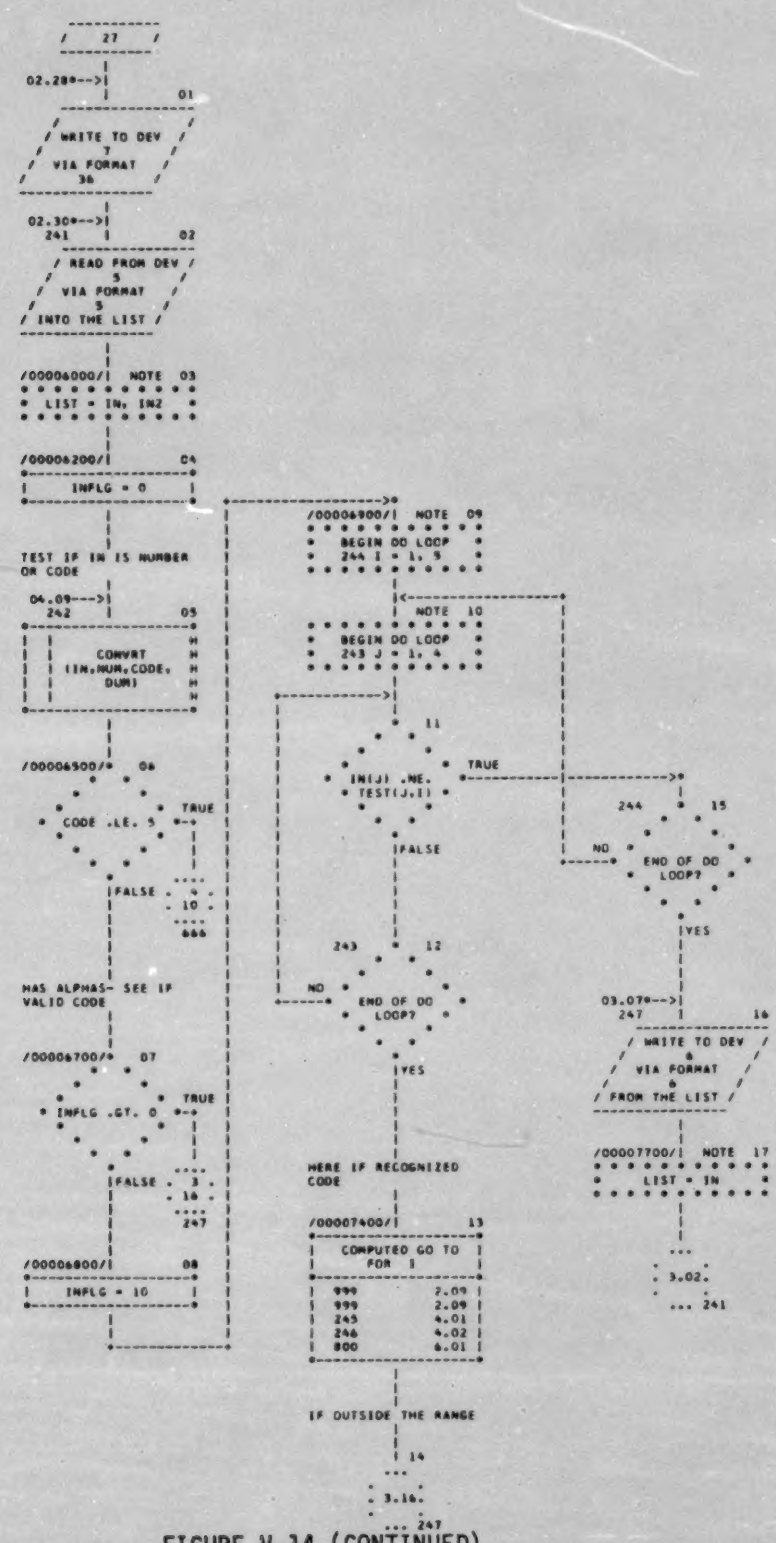


FIGURE V-14 (CONTINUED)

CHART TITLE - PROCEDURES

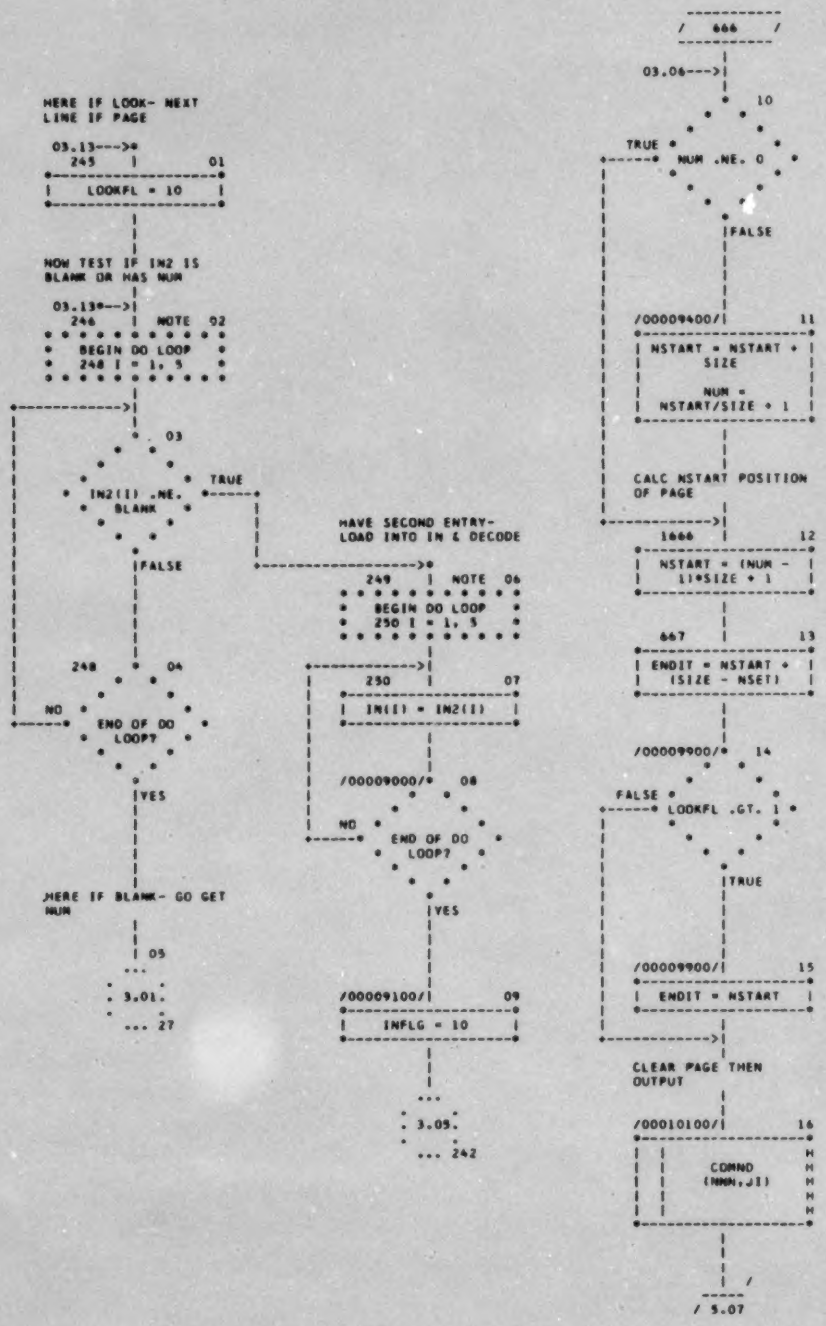


FIGURE V-14 (CONTINUED)

CHART TITLE - PROCEDURES

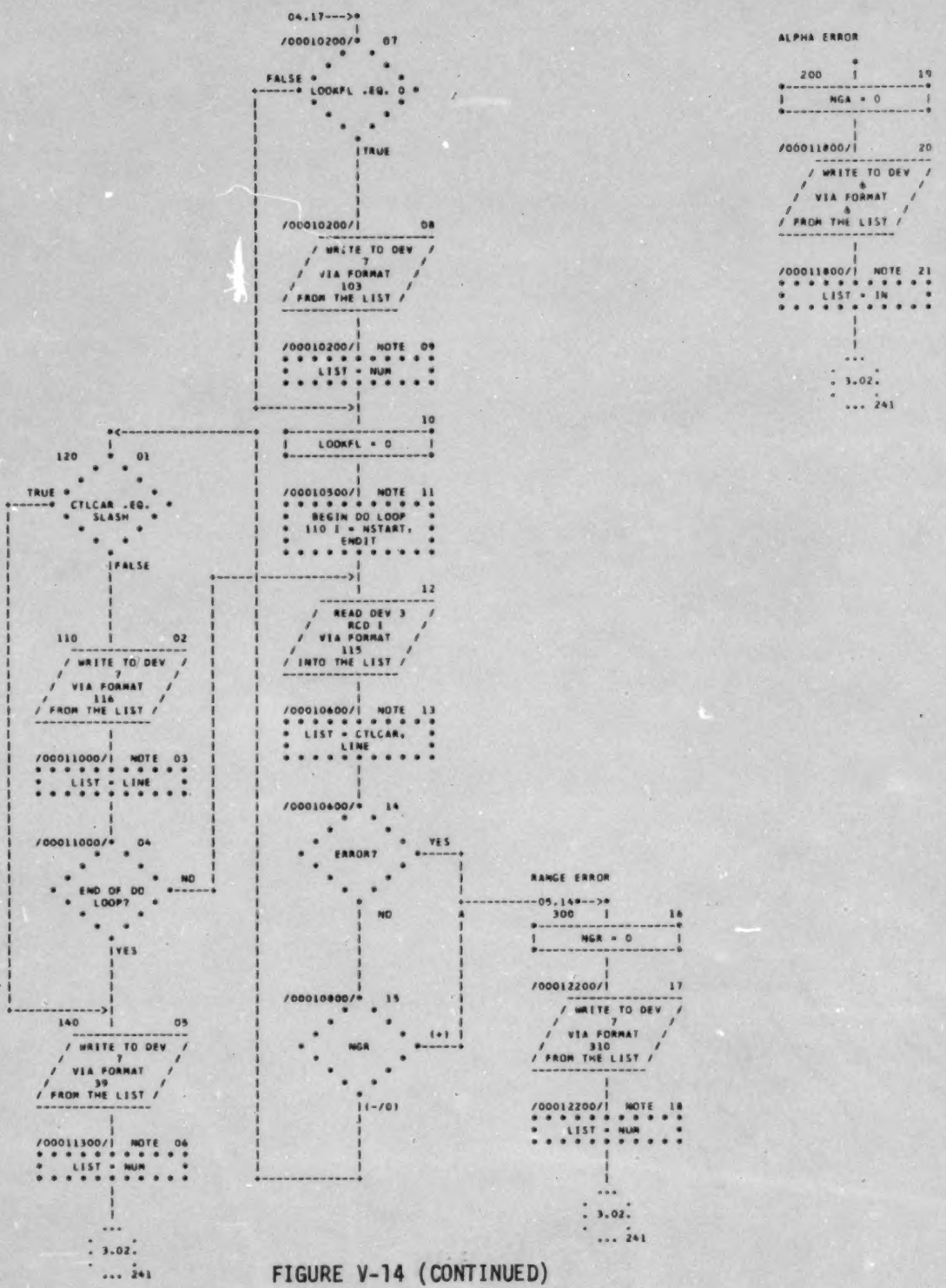


FIGURE V-14 (CONTINUED)

CHART TITLE - PROCEDURES

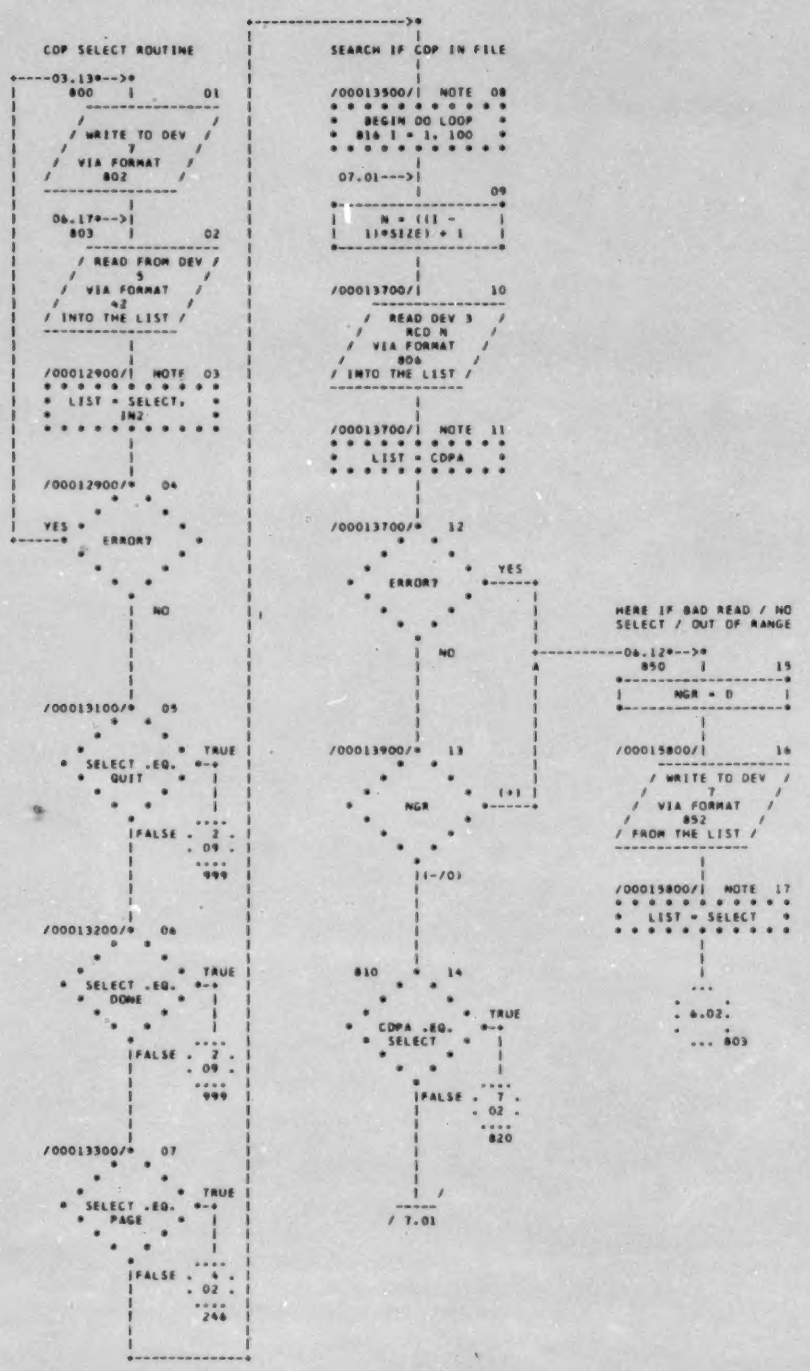


FIGURE V-14 (CONTINUED)

CHART TITLE - PROCEDURES

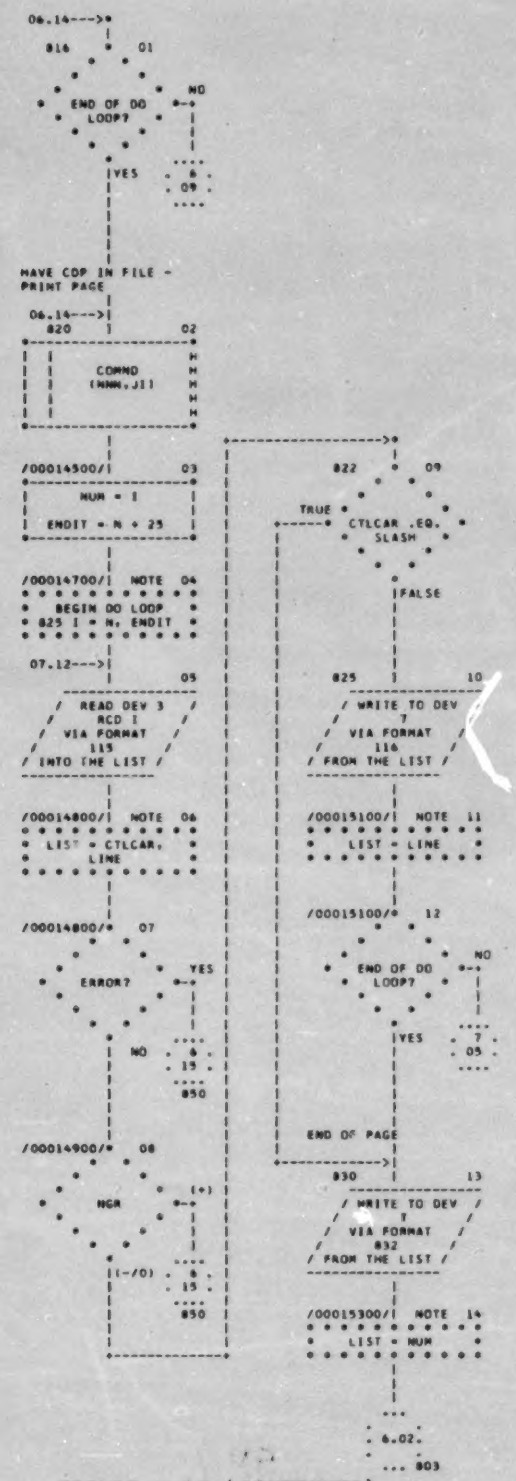


FIGURE V-14 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      INTEGER*2 IN(5), IN2(5), CODE, TEST(4,5), SIZE,NSTART,NNN,ENDIT
      DATA TEST/'Q','U','I','T','D','O','N','E','L','O','O','K',
       'P','A','G','E','C','D','P',' '/
      INTEGER*2 BLANK /' '/
      REAL*4 CDP, DONE, PAGE, QUIT, SELECT
      REAL*8 LINE (10)
      INTEGER*4 NUM
      COMMON NGA,NGR
      EXTERNAL RANGE
      EXTERNAL ALPHA
      DEFINE FILE 3 (2500,81,E,J3)
      DATA SLASH/'/'/,CDP/'CDP '/,DONE/'DONE'/,PAGE/'PAGE'/,QUIT/'QUIT'/
15    FORMAT (A1,80X)
25    FORMAT (' NON STANDARD FILE TYPE - DEFAULT 20 LINES PER PAGE')
35    FORMAT (' STANDARD FILE - ',I2,' LINES')
10028 FORMAT (1X, I3, ' PAGES IN FILE')
40    FORMAT (/' +COURSE FILE - SELECT BY ''CDP'' OR ''PAGE'' ')
36    FORMAT (/' +ENTER PAGE NUMBER')
5     FORMAT (10A1)
6     FORMAT (1X,5A1,' +INVALID- RETRY'/)
103   FORMAT (' PAGE', I4)
115   FORMAT (A1,9A8,A7)
116   FORMAT (1X,9A8,A7)
39    FORMAT (' PAGE', I4,'  +PAUSING- ENTER NEXT PAGE NO.')
310   FORMAT (1X,I4,1X,'OUT OF RANGE - RETRY')
802   FORMAT (/' +ENTER CDP')
42    FORMAT (A4,1X,5A1)
806   FORMAT (1X, A4)
832   FORMAT (' PAGE',I4,'  +PAUSING- ENTER CDP, PAGE, OR QUIT')
852   FORMAT (1X, A4, ' +NOT IN FILE OR INVALID - RETRY')
998   FORMAT (' +DISPLAY TPF ENDED')
```

FIGURE V-14 (CONTINUED)

08/14/75 AUTOFLOW CHART SET - FINAL -DISPLAY- FLOWCHARTS PAGE 09

CHART TITLE - SUBROUTINE RANGE(KK,LL)

```
        ----------
       / RANGE   /
       ----------
            |
            |
            |          01
*-------------------*
|     NGR = 10      |
|                   |
|      KK = 0       |
|                   |
|     LL = 232      |
*-------------------*
            |
            |
/00017100/| 02
        ---------
       * EXIT   *
        ---------
```

08/14/75 AUTOFLOW CHART SET - FINAL -DISPLAY- FLOWCHARTS PAGE 10

CHART TITLE - NON-PROCEDURAL STATEMENTS

COMMON NGA,NGR

08/14/75 AUTOFLOW CHART SET - FINAL -DISPLAY- FLOWCHARTS PAGE 11

CHART TITLE - SUBROUTINE ALPHA(KK,LL)

```
        ----------
       / ALPHA   /
       ----------
            |
            |
            |          01
*-------------------*
|     NGA = 10      |
|                   |
|      KK = 0       |
|                   |
|     LL = 215      |
*-------------------*
            |
            |
/00017800/| 02
        ---------
       * EXIT   *
        ---------
```

08/14/75 AUTOFLOW CHART SET - FINAL -DISPLAY- FLOWCHARTS PAGE 12

CHART TITLE - NON-PROCEDURAL STATEMENTS

COMMON NGA,NGR

FIGURE V-14 (CONTINUED)

TPF LOGIC DESIGN

The design of the TPF model is such that it might be termed a front-end loaded model. That is, all schedule and demand conditions for a course are loaded at the beginning of the year, and the model then projects the resultant conditions for the remainder of the year. Likewise, the basic model projects, rather than predicts. The model itself is a true simulation, or mathematical representation of the mechanics of the overall scheduling function for the center. All major functions that affect student flow are modeled to a detail similar to the actual operations at the center, which include quota control, BUPERS detailing, scheduling, failures, and setbacks. The model outputs, however, can be predictive if the inputs themselves are predictive. To use the model for decision-making purposes, the user must input his predictions of demands, schedule changes, new courses, etc., and the model then projects the annual throughputs, failures, AOB's, and such, based on these predictions. Likewise, a simulation model is not an optimization model; the large number of variable input parameters precludes this. The user can manipulate the model inputs to achieve optimal results.

PROGRAM LOGIC DESIGN. The TPF model consists of a main control program and seven subroutines, all coded in the FORTRAN language. This structure was chosen for several reasons. All input data enter through the main control program, which edits the input data and builds a matrix for communication between the subroutines. Thus, if input methods should be changed, such as entering backlog as students rather than the present method of entering weeks, only one routine would be required. As another example, the entire scheduling routine is contained in subroutine CRSRLS. This subroutine represents the methods and procedures currently in use at the center. Should it be desirable to represent other centers with different methods and procedures, a new subroutine to represent that center could be written, and the appropriate subroutine could be linked to the model at execute time. In summary, the model has been written in a modular form so as to allow adaptability and ease of modification without extensive rewrite.

Certain programming conventions are followed throughout the design of the TPF model and its subroutines. All input data are read in at the beginning of the operation, with proper editing and error checking provided at that time. Parameter information is read in and various flags are set to control subsequent execution. In both cases, assumptions are made for missing or inaccurate data and processing continues with appropriate error messages. Finally, all communication throughout the model is through COMMON storage, with minimal use made of subroutine parameter passing.

TPF MODEL MAIN. The TPF model is compiled and stored for execution under the name KEND. This is to allow executives to use the name TPF. The basic control routine in this FORTRAN program is called MAIN. This module is the primary interface of the TPF model with the rest of the TPF system All input data and parameters are entered and edited by this section, and all control of the TPF model execution is set up by MAIN. MAIN also controls all cycling for the various loops necessary for the TPF simulation.

Parameter Options. The first operation in MAIN is to read in the control parameter card. This method of control was chosen to provide for card batch

operation as well as on-line terminal execution. The format of the parameter card is shown in Figure V-3C. Four elements of the parameter card are key to execution; input, first year, last year, and AOB constant, as they set up control for execution. Input specifies whether the data are to be read from cards(05), or from an on-line data set (10), with the default value being 10. First year and last year are used to control the years of the projection (default 1 year, maximum 3 years) and the years entered are printed as titles on output reports. They do not currently control data base access. AOB constant is used to denote the method and value to be used for average-on-board calculations. A value of 300 or greater will cause weekend days to be counted in AOB calculations, while a value of less than 300 will alter the model so that only working days are counted.

The next 8 parameters on the parameter card control across-the-board modification. The first character in each field is a flag (0 for off, 1 for active) and the remaining four characters are a multiplier. If the flag is 1 or greater, the multiplier is passed to other routines for data overrides. The next two fields allow the user to select only a course CDP number or school for execution, without the need to alter the data base.

The last 10 fields on this card contain flags used to control various options of the TPF model. If the INIT flag is set, the model will perform the pre-run initialization described later in this section. If the print flag is 0, only on-line terminal output is provided, a 2 in this field providing only off-line printer output, while a 1 produces both forms of output. The output data provided is controlled by flags COURSE($R1), SCHOOL($R2), and CENTER($R3). A 1 in any of these fields will produce the specified reports, while all zeros in these fields will cause the model to cycle with no output data generated. The rest of the flags control trace routines used for detailed analysis of operation of each of the TPF model subroutines. Use of these trace flags is not recommended for normal operation due to the volume produced. They may be set, however, using program PARAM and their names may be displayed by the program by requesting FLAGS.

Data Entry and Editing. The next major section of program MAIN consists of the logic to read data into storage, provide reasonableness checks, and to perform data overrides. All data are read into main storage to reduce disk I/O operations as the system runs under virtual storage. Each data element is tested for reasonableness and the value is clamped if necessary, allowing the model to execute using a data base with known errors. The various overrides are performed next for those fields where the override flag is set. Logic is included to assure that the overrides produce meaningful results, i.e., course convenings clamped to integer values, rather than allowing a value such as 2.87 convenings. Following these operations, MAIN passes control to the cycling portion of the model.

Model Control and Cycling. Except for one routine, the test of the main program logic is straightforward, using standard FORTRAN coding techniques. Three major cycling paths exist in the model; the year loop, the quarter loop, and the week loop. One of these, the quarterly loop, can undergo alteration to initialize the model. The TPF model does not snapshot existing conditions, but

rather it views conditions over a period of time. It considers the schedule and classes as they are today, and projects for one year. One important factor must be recognized. Today there may be 70 courses in session that were started in a preceding period and which, because of their length, terminate in the period being projected. To represent this, the quarter loop can reconfigure and, for the first quarter loop, the model operates as though it were the quarter prior to model start. No totals are accumulated for the run; the only effect is to release courses so that they may be active at model start. Subroutine PRERUN is used to accomplish this, and this initialization will be discussed in further detail in that segment.

The lowest order cycling loop in the model is the week loop. An understanding of how this loop operates will probably lead to an understanding of the entire model. This loop first scans each course loaded in the model, one at a time, to ascertain if a course should convene that week. It does this by testing the convening frequency of the course against a matrix loaded in core, for convening frequencies of 1 to 50 convenings per year. Before this comparison is made, the matrix is rationalized for offset, change week, and multiple convenings. This routine is described immediately following this segment, but for now, all that is necessary to know is that if this matrix returns a code that the course should start, subroutine PRERUN will be called and the course started. When a course is started, it is placed in an active matrix, which is described in the subroutine PRERUN segment. Upon completion of the test for possible convening of all courses, the MAIN program then tests all active courses to see if any end in that week. As the model cycles on a weekly basis, many courses will begin and end in the same weekly loop. When a course is found to end, subroutine CRSEND is called which closes out the course, allocates the AOB, and calculates attrition. When all of the 300 possible active courses have been tested, the weekly loop recycles to run another week.

Every 13 weeks, the quarter loop cycles, calling subroutine QRTEND, to allocate AOB against quarters and accumulate totals. Finally, after 4 quarters, the year loop cycles, calling subroutine YEARND, which controls printout for the yearly report.

Scheduling Algorithm. Several scheduling algorithms were studied as candidates for this model. As part of the design effort, the actual convening dates for many of the courses were plotted. These plots were then analyzed and an allowance was made for common holidays. The data were then transferred to the matrix shown in Figure V-15. This shows the scheduled convening weeks for courses with a convening frequency of 1 to 50 per year. Many courses convened more frequently that this, and these are handled by breaking them into multiple courses, each with convening frequencies of less than 50, but whose overall convenings are the same as the original. For example, a course with 51 convenings is handled as two courses, one that convenes 25 times annually. Provision exists to break each course into 10 separate courses. The convening matrix itself is stored in core and names ISCHED. The data are loaded in hexidecimal format to accomplish this. FORTRAN allows the loading of hexidecimal data, but does not allow individual bit testing or manipulating. Therefore, a synthetic method of bit testing was developed. This technique involves extracting the proper bit pattern by division by various constants based on week and quarter, which leaves a 1 on the low order bit position if the course is to start, or 0 if it should not. This extracted bit pattern is then tested to determine whether the low order bit is a 0 or 1, which is accomplished by a division by integer 2, and then a multiplication by integer 2. If the result

is the same as the original, the low order bit was 0 and the course is not released. Two rationalizations must be made prior to lookup in this table. First, if a course convenes only once a year, the matrix will start it the first week of the model run. In reality, this course may convene on the 21st week. Offset was introduced as an input to the model to allow fine tuning of the course release date. An offset of 20, for example, will tell the model to enter the schedule matrix as week 1 when the model is in the 21st week, the second term to be rationalized. If the schedule change date has passed, the "B" convening schedule is used, otherwise the "A" convenings are used.

As this technique uses matrix lookup, the table must be protected for invalid entries while preserving flexibility. If the convening frequency is 0, the matrix will be bypassed. If the week is greater than 52, 52 is subtracted from the week until it is less than 52. If offset is positive, such as 21, no courses will be released until the 22nd week, regardless of the convening frequency. A negative offset of 31 would accomplish the same schedule shift, and courses could be released from the start of the model run.

SUBROUTINE CRSRLS. The course release subroutine contains the scheduling logic for the center. This includes keeping backlog counts, scheduling only local seats, and overbooking certain courses with high no-show rates. The logic of this subroutine schedules the BUPERS students based on the annual BUPERS plan, divided by current convening frequency. This is done even if the result exceeds the seats controlled by BUPERS. Next, local quotas are released. This is accomplished by releasing students to a backlog pool based on local demand, divided by current convenings. This pool is released to the course up to the local capacity. No-shows are then calculated using historical no-show percentages. Only local students are used for this calculation as BUPERS students generally are on TAD or PCS orders, and are not identifiable as no-shows. Finally, for courses running near capacity, up to 10 percent of the capacity is released from the backlog pool as substitute quotas. This routine then builds an entry in the course active matrix. Included is utilization, weeks till termination, and a term to allocate AOB between quarters and between classes if a student should be set back. This subroutine closely represents the scheduling methods in use at the center. If other centers, with different algorithms, should be studied, this subroutine would most likely be replaced by one representing the location under study.

SUBROUTINE PRERUN. As mentioned previously, the model can run in an initialization mode for one quarter prior to the actual model run. Subroutine PRERUN is called by CRSRLS any time the model is in the initialization mode. It is required for two functions. First, courses are tested prior to release to ascertain whether they should be running at model start or not at all. Secondly, this subroutine allocates AOB for those courses to assure that AOB applicable to the first quarter is charged to the first quarter.

SUBROUTINE CRSEND. This subroutine is called each time a course is found to end in a particular week. Several functions are performed by this subroutine. The first function consists of disposing of all references to the course in the active matrix. Secondly, failures are calculated, with fractional failures pooled, using the current failure rate. The third function is to calculate non-academic disenrollments, based on historical rates. This subroutine also

calculates setbacks. Setback logic currently allows setbacks for those courses 3 weeks or longer, and where more than 1 convening is in session at a time. This setback rate is currently based on a fixed percentage of failures per class. AOB for setback is allocated between the "entered in" and "finished in" convening. Finally, this subroutine calculates AOB based on the current calculation method in effect.

SUBROUTINE QTREND. This short subroutine is used to accurately allocate AOB between quarters. Only the active courses are tested at the end of a quarter, the accumulated AOB is charged to the current quarter, and the active course AOB is adjusted accordingly.

SUBROUTINE YEARND. The function of subroutine YEARND is to calculate annual totals for all courses, prepare the model for multiple year runs, aggregate school and center total if those reports are keyed, and call appropriate subroutines to provide output reports. This subroutine requires that courses be entered in the data matrix sorted by school code if proper school summary reports be produced.

SUBROUTINE CRTOUT. Subroutine CRTOUT produces course, school, and center reports in a 79 column format. This format is compatible with the Hazeltine 2000 terminal which only has 79 usable columns for formatted data. A sample of the reports produced by subroutine CRTOUT is shown in Figure V-16. The format used for the control character in column 1 was described in this section under display support programs. The same format routines are used for the data portion of the school and center reports, and subroutine YEARND loads center data into the school matrix prior to calling CRTOUT to provide a center report.

SUBROUTINE NPRINT. Subroutine NPRINT produces course, school, and center reports in a 130 column format compatible with line printers or the 2741 terminal. As this routine is designed to interface directly with a line printer, standard carriage control conventions are utilized. Also, three course reports are printed per page, and page numbering is provided. The data presented are identical to those produced by CRTOUT.

COURSE SCHEDULER MATRIX

TPF MODEL COURSE RELEASE SCHEDULER

ANNUAL CONV	*** 1ST QTR *** (WKS 1-13)	*** 2ND QTR *** (WKS 1-13)	*** 3RD QTR *** (WKS 1-13)	*** 4TH QTR *** (WKS 1-13)
1	X ---- ---- ----	- ---- ---- ----	- ---- ---- ----	- ---- ---- ----
2	X ---- ---- ----	- ---- ---- ----	- X--- ---- ----	- ---- ---- ----
3	X ---- ---- ----	- ---X ---- ----	- ---- ---- -X--	- ---- ---- ----
4	X ---- ---- ----	X ---- ---- ----	- X--- ---- ----	X ---- ---- ----
5	X ---- ---- -X--	- ---- --X- ----	- ---- -X-- ----	- --X- ---- ----
6	X ---- ---- X---	- ---X ---- ----	- X--- ---- -X--	- ---- X--- ----
7	- X--- ---X ----	- -X-- ---- X---	- ---- X--- ---X	- ---- -X-- ----
8	X ---- --X- ----	X ---- -X-- ----	- X--- ---X ----	- X--- --X- ----
9	- X--- -X-- ---X	- ---X ---- -X--	- ---X ---- -X--	- -X-- ---X ----
10	X ---- X--- -X--	- -X-- --X- ----	- X--- -X-- --X-	- --X- ---X ----
11	X ---- X--- -X--	- -X-- -X-- -X--	- ---X ---- X---	- X--- X--- X---
12	- X--- X--- X---	X ---X ---X ----	- X--- -X-- -X--	- X--- X--- X---
13	- X--- X--- X--X	- --X- --X- --X-	- ---X ---X --X-	- -X-- -X-- -X--
14	X ---X ---X --X-	- -X-- X--- X---	- -X-- X--- X--X	- --X- -X-- -X--
15	X --X- --X- -X--	X ---X --X- -X--	- --X- -X-- X---	X --X- -X-- -X--
16	X --X- --X- -X--	X --X- -X-- X---	- X--- X--X --X-	- X--X --X- -X--
17	- X--X --X- -X-X	- -X-- X--X --X-	- --X- -X-- X--X	- -X-- X--X --X-
18	X --X- -X-- X--X	- -X-- X-X- -X--	- -X-- X--X --X-	- X-X- -X-- X-X-
19	- X-X- X--- X-X-	- X-X- -X-X --X-	- --X- -X-X --X-	- X-X- -X-- X--X
20	X --X- X--X -X--	X -X-- X-X- -X--	- X--X -X-- X-X-	- X-X- -X-X --X-
21	- X-X- X--X -X-X	- -X-X -X-- X-X-	- -X-- X-X- -X-X	- X--X -X-X --X-
22	- X-X- X--X -X-X	- -X-X -X-X -X--	- -X-X -X-X -X-X	- -X-X --X- X-X-
23	- X-X- X-X- X-X-	- X-X- X-X- X-X-	- ---X X-X- X-X-	- X-X- X-X- X-X-
24	- X-X- X-X- X-X-	X -X-X -X-X -X--	- -X-X -X-X -X-X	- X-X- X-X- X-X-
25	X -X-X -X-X -X-X	- X-X- X-X- X-X-	- -X-X -X-X -X-X	- X-X- X-X- X-X-
26	X X-X- X-X- X-X-	X -X-X -X-X -X--	- X-X- X-X- X-X-	X -X-X -X-X -X-X
27	X X-X- X-X- X-X-	X -X-X -XX- X-X-	- X-X- X-XX -X-X	- X-X- X-X- X-X-
28	X -X-X -XX- X-X-	X X-X- X-X- X-X-	- X-X- X-X- X-X-	X X-X- XX-X -X-X
29	X -X-X -XX- X-X-	X X-X- X-XX -X--	- X-XX -X-X X-X-	X -XX- X-X- XX-X
30	- XX-X -XX- X-XX	- X-XX -X-X X-X-	- -XX- X-XX -X-X	X -X-X X-X- XX-X
31	X -X-X X-XX -X-X	X -X-X X-X- XX--	- XX-X X-X- XX-X	X -X-X X-XX -XX-
32	- XX-X X-XX -XX-	X X-XX -X-X X-X-	- X-XX -XX- XX-X	X -X-X X-XX -XX-
33	X -XX- XX-X X-X-	X X-XX -XX- XX--	- XX-X X-XX -XX-	X X-XX -XX- XX-X
34	- XX-X XX-X X-XX	- XX-X X-XX -XX-	- -XXX -XX- XX-X	X -XX- XX-X X-XX
35	- XX-X XX-X X-XX	- XXX- XX-X X-X-	- XX-X X-XX -XXX	- XX-X X-XX X-XX
36	- XXX- XXX- XX-X	X X-XX -XXX -XX-	- X-XX -XXX -XX-	X XX-X X-XX X-XX
37	X -XXX -XXX -XX-	X XX-X XX-X XX--	- XXX- XXX- XX-X	X X-XX X-XX X-XX
38	X -XXX -XXX -XXX	- XXX- XXX- XXX-	- -XXX X-XX X-XX	X -XXX -XXX -XXX
39	- XXXX -XXX X-XX	X X-XX X-XX X-X-	- XXX- XXXX -XXX	X -XXX -XXX -XXX
40	- XXXX -XXX X-XX	X X-XX XX-X XXX-	- -XXX X-XX XX-X	X XX-X XXX- XXXX
41	X -XXX X-XX XXX-	X XXX- XXXX X-X-	- XXX- XXXX X-XX	X X-XX XXX- XXXX
42	- XXXX XX-X XXXX	- XXXX X-XX XXX-	- -XXX XXX- XXXX	X -XXX XX-X XXXX
43	X -XXX XXX- XXXX	X X-XX XXXX -XX-	- XXXX -XXX XXX-	X XXXX X-XX XXXX
44	- XXXX XXXX -XXX	X XXX- XXXX XXX-	- XXXX -XXX X-XX	X XXXX -XXX XXXX
45	- XXXX XXXX X-XX	X XXXX XX-X XXX-	- XXXX X-XX XXXX	X XX-X XXXX XXXX
46	- XXXX XXXX XXXX	- XXXX XXXX XXX-	- -XXX XXXX XXXX	- XXXX XXXX XXXX
47	- XXXX XXXX XXXX	X XXX- XXXX XXX-	- XXXX XXXX X-XX	X XXXX XXXX XXXX
48	- XXXX XXXX XXXX	X XXXX XXXX XXX-	- -XXX XXXX XXXX	X XXXX XXXX XXXX
49	- XXXX XXXX XXXX	X XXXX XXXX XXX-	- XXXX XXXX XXXX	X XXXX XXXX XXXX
50	X XXXX XXXX XXXX	X XXXX XXXX XXX-	- XXXX XXXX XXXX	X XXXX XXXX XXXX

FOR EACH WEEK IN THIS SCHEDULE: AN 'X' INDICATES THE COURSE WILL BE RELEASED
A '-' INDICATES THE COURSE WILL NOT BE RELEASED

NOTE: IF ANNUAL CONVENINGS FOR A COURSE IS GREATER THAN 50, THE COURSE WILL BE TREATED AS MULTIPLE COURSES, EACH HAVING 50 OR LESS CONVENINGS PER YEAR.

FIGURE V-15

Ø284 FINANCIAL MGMT

SCHL JNO CATALOG NO NEC
S/DA 6552 A-551-ØØ56

P AOB	PFAIL	TYPE	PRTY
2.Ø	7.Ø%	F1	31

--SCHEDULE DATA--	FY	FY+1	FY+2
ANNUAL CONVENINGS	13	Ø	Ø
COURSE LENGTH DYS	5.Ø	Ø.Ø	Ø.Ø
COURSE CAPACITY	25	Ø	Ø
BUPERS CAPACITY	Ø	Ø	Ø
ANNUAL DEMAND	15Ø	18Ø	18Ø
BUPERS DEMAND	Ø	Ø	Ø

CRS BACKLOG	
# STU	32
# WKS	5
FY+1 WK	Ø
FY+2 WK	Ø
OFFSET	Ø

QUARTERLY PROJECTIONS

	1 QR FY 75		2 QR FY 75		3 QR FY 75		4 QR FY 75		ANNUAL TOTAL	
# CONVENINGS	4		3		3		3		13	
CRS CAPACITY	1ØØ		75		75		75		325	
UTILIZATION	67	67.Ø%	3Ø	4Ø.Ø%	29	38.7%	3Ø	4Ø.Ø%	156	48.Ø%
NO SHOWS	11	14.1%	5	14.3%	5	14.7%	5	14.3%	26	14.3%
BAKLOG #/WKS	Ø	Ø WKS	Ø	Ø WKS	Ø	Ø WKS	Ø	Ø WKS	Ø	Ø WKS
STU DAYS/AOB	335	3.7	15Ø	1.6	145	1.6	15Ø	1.6	78Ø	2.1
TOTAL PASS	62	92.5%	27	9Ø.Ø%	25	86.2%	26	86.7%	14Ø	89.7%
TOTAL FAIL	5	7.5%	3	1Ø.Ø%	3	1Ø.3%	3	1Ø.Ø%	14	9.Ø%
SETBACK ACAD	Ø	Ø.Ø%	Ø	Ø.Ø%	Ø	Ø.Ø%	Ø	Ø.Ø%	Ø	Ø.Ø%
DISENROLL NA	Ø	Ø.Ø%	Ø	Ø.Ø%	1	3.4%	1	3.3%	2	1.3%

TPF MODEL -- FISCAL YEAR 1976 SUMMARY
SCHOOL CODE COMM

COMM SCHOOL SUMMARY

	1 QR FY 76		2 QR FY 76		3 QR FY 76		4 QR FY 76		ANNUAL TOTAL	
# CONVENINGS	47		41		44		41		173	
CRS CAPACITY	683		589		645		589		25Ø6	
UTILIZATION	6Ø2	88.1%	447	75.9%	441	68.4%	383	65.Ø%	1873	74.7%
NO SHOWS	77	11.3%	68	13.2%	64	12.7%	58	13.2%	267	12.5%
BAKLOG #/WKS	181	7 WKS	9Ø	6 WKS	46	5 WKS	27	3 WKS	27	3 WKS
STU DAYS/AOB	281Ø	3Ø.8	2Ø52	22.5	2Ø51	22.5	1661	18.2	8574	23.5
TOTAL PASS	588	97.7%	434	97.1%	43Ø	97.5%	372	97.1%	1824	97.4%
TOTAL FAIL	14	2.3%	13	2.9%	11	2.5%	11	2.9%	49	2.6%
SETBACK ACAD	Ø	Ø.Ø%	Ø	Ø.Ø%	Ø	Ø.Ø%	Ø	Ø.Ø%	Ø	Ø.Ø%
DISENROLL NA	Ø	Ø.Ø%	Ø	Ø.Ø%	Ø	Ø.Ø%	Ø	Ø.Ø%	Ø	Ø.Ø%

TPF MODEL -- FISCAL YEAR 1976 SUMMARY
** TOTALS FOR FLEET TRAINING CENTER **

FTC - NORVA SUMMARY

	1 QR FY 76		2 QR FY 76		3 QR FY 76		4 QR FY 76		ANNUAL TOTAL	
# CONVENINGS	759		695		678		72Ø		2852	
CRS CAPACITY	239Ø8		22514		21648		24265		92335	
UTILIZATION	162Ø9	67.8%	12559	55.8%	11825	54.[illegible]%	12916	53.2%	53511	58.Ø%
NO SHOWS	2423	13.Ø%	1974	13.6%	1861	13.[illegible]%	2Ø82	13.9%	834Ø	13.5%
BAKLOG #/WKS	3432	9 WKS	2676	11 WKS	2325	13 WKS	2172	15 WKS	2172	15 WKS
STU DAYS/AOB	92Ø66	1ØØ8.9	73855	8Ø9.4	7152Ø	787.8	73931	81Ø.2	311372	853.1
TOTAL PASS	15686	96.8%	12Ø69	96.1%	11348	96.Ø%	12393	95.9%	51496	96.2%
TOTAL FAIL	417	2.6%	375	3.Ø%	365	3.1%	4ØØ	3.1%	1557	2.9%
SETBACK ACAD	36	Ø.2%	27	Ø.2%	27	Ø.2%	3Ø	Ø.2%	12Ø	Ø.2%
DISENROLL NA	1Ø6	Ø.7%	115	Ø.9%	112	Ø.9%	125	1.Ø%	458	Ø.9%

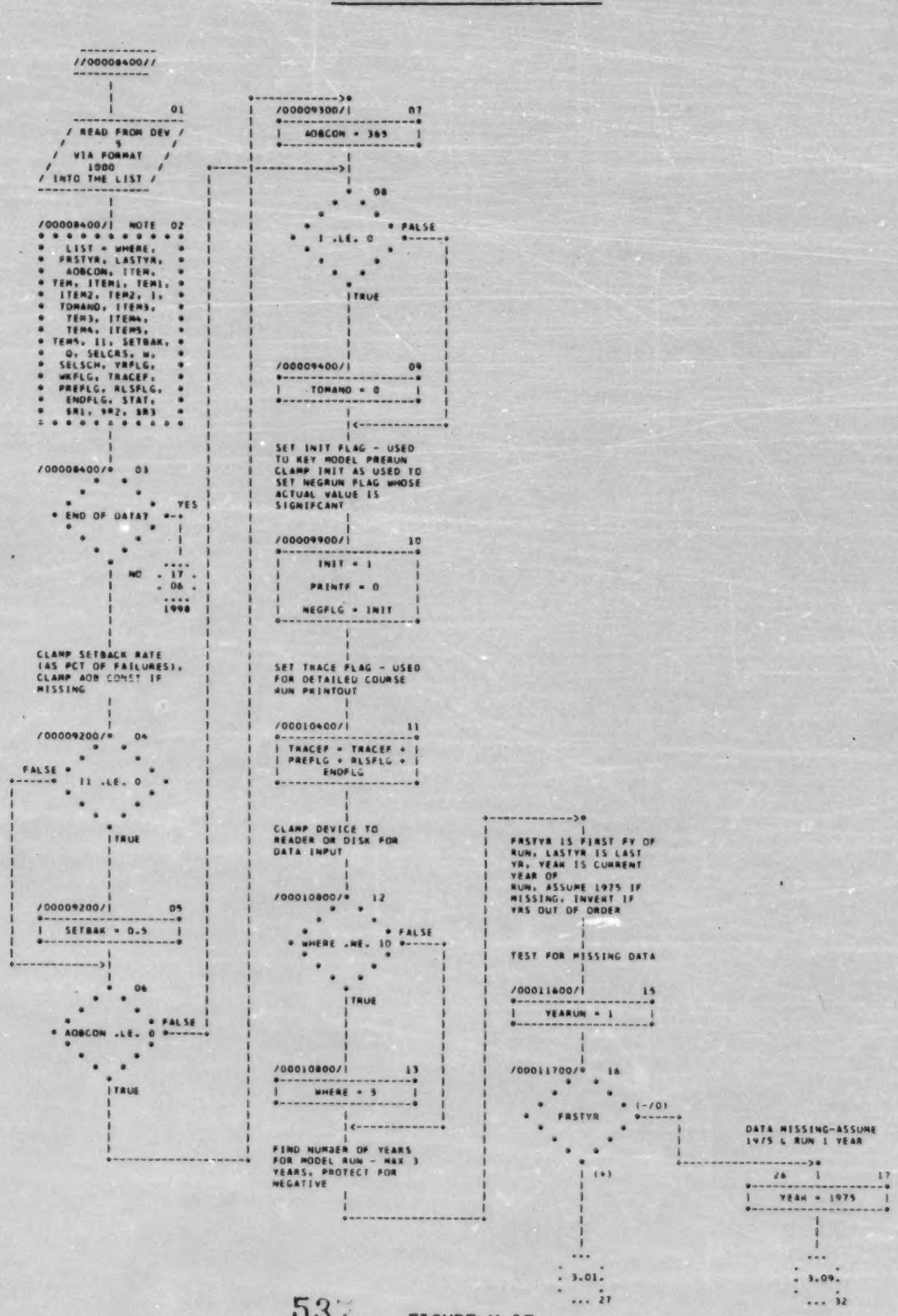
SAMPLE TPF OUTPUTS

FIGURE V-16

538

CHART TITLE - PROCEDURES

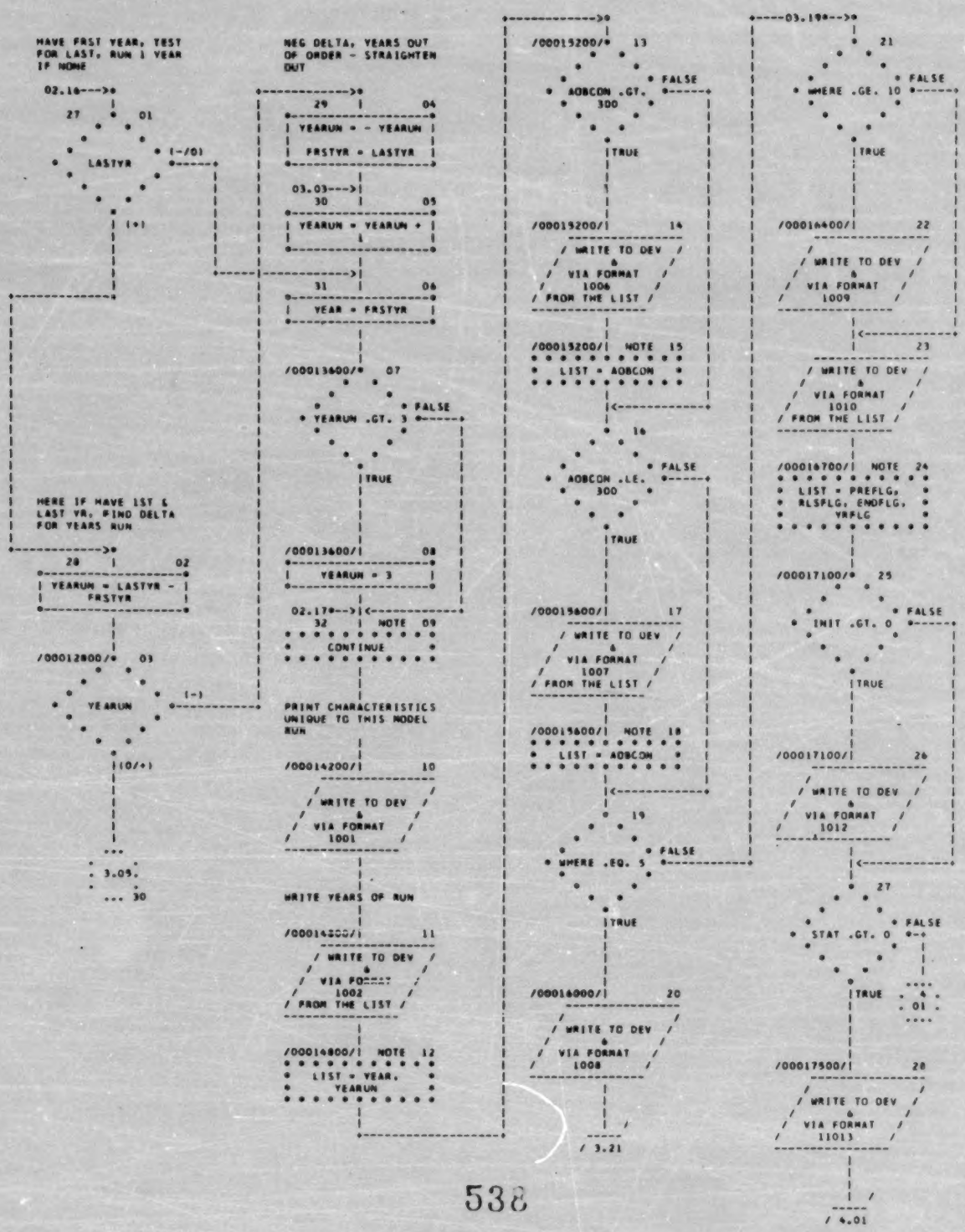
AUTOFLOW - PROGRAM KEND



537

FIGURE V-17

CHART TITLE - PROCEDURES



538

FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

```
 03.27*-->*
          |
END OF MODEL RUN DATA
MESSAGES - OUTLINE
ERROR MESSAGES
          |
          |
/00018100/|        01
 ----------------
   /                /
  / WRITE TO DEV  /
 /      6        /
/  VIA FORMAT   /
/     1011     /
----------------
          |
          |
          |
/00018500/*   02
        *   *
      *       *
 FALSE *       *
+-----*  Q .GT. 0   *
|     *           *
|       *       *
|         *   *
|           *
|           |TRUE
|           |
|           |
|           |
|           |
/00018500/|        03
 ----------------
   / WRITE TO DEV  /
  /      6        /
 /  VIA FORMAT   /
/     12014     /
/ FROM THE LIST /
----------------
          |
          |
/00018500/|  NOTE  04
* * * * * * * * * * *
*  LIST = SELCRS    *
* * * * * * * * * * *
          |
+---------->|
          |
          *   05
        *   *
      *       *
 FALSE *       *
+-----*  W .GT. 0   *
|     *           *
|       *       *
|         *   *
|           *
|           |TRUE
|           |
|           |
|           |
|           |
/00018600/|        06
 ----------------
   / WRITE TO DEV  /
  /      6        /
 /  VIA FORMAT   /
/     12014     /
/ FROM THE LIST /
----------------
          |
          |
/00018600/|  NOTE  07
* * * * * * * * * * *
*  LIST = SELSCH    *
* * * * * * * * * * *
          |
+---------->|
          |
READ COURSE DATA FROM
DATA BASE INTERFACE
INTO MATRIX
COURSE MATRIX SET UP
FOR MAXIMUM OF 176
INDIVIDUAL COURSES
CRSES EQUALS NUMBER
OF UNIQUE COURSES FOR
THIS RUN
          |
          |
          +-------------*
```

```
+------------------>*
/00019300/|        09
*-------------------*
|     CRSES = 0     |
*-------------------*
          |
 04.18*-->|
          |        10
*-------------------*
|   I = CRSES + 1   |
*-------------------*
          |
          |
/00019600/|        11
 ----------------
   / READ FROM DEV /
  /     WHERE     /
 /  VIA FORMAT   /
/      10       /
/ INTO THE LIST /
----------------
          |
          |
/00019600/|  NOTE  12
* * * * * * * * * * *
*  LIST = COP(I),   *
*    CATNO1(I),     *
*    CATNO2(I),     *
*     NAME1(I),     *
*     NAME2(I),     *
*    CONV(1,I),     *
*     LEN(1,I),     *
*     CAP(1,I),     *
*    BCAP(1,I),     *
*   DEMAND(1,I),    *
*   BDMAND(1,I),    *
*    OFFSET(I),     *
*     FAIL(I),      *
*     NADIS(I),     *
* NOSHOW(I), DUMMY, *
*    BAKLOG(I),     *
*      JNO(I),      *
* * * * * * * * * * *
          |
/00019600/|  NOTE  13
* * * * * * * * * * *
*    CONV(2,I),     *
*     LEN(2,I),     *
*     CAP(2,I),     *
*    BCAP(2,I),     *
*   DEMAND(2,I) ,   *
*   BDMAND(2,I),    *
*    CHGWK(1,I),    *
*    CONV(3,I),     *
*     LEN(3,I),     *
*     CAP(3,I),     *
*    BCAP(3,I),     *
*   DEMAND(3,I),    *
*   BDMAND(3,I),    *
*    CHGWK(2,I),    *
*     PAOB(I),      *
*     PFAIL(I),     *
* PRTY(I), TYPE(I), *
* * * * * * * * * * *
          |
          |
          |  NOTE  14
* * * * * * * * * * *
*     PNEC(I),      *
*     SCHOOL(I)     *
* * * * * * * * * * *
          |
          |
          |
/00019600/*   15
        *   *
      *       *
    *           *   YES
  * END OF DATA?  *-->
    *           *   |
      *       *     |
        *   *       |
          *         |
          |  NO  ----
          |     . 5 .
          |     . 01 .
          |      ----
          |        2
          |
          |
          |
COURSE / SCHOOL
SELECT ROUTINE (Q IS
COURSE FLAG, W IS
SCHOOL FLAG)
          |
          |
          +-------------*
```

```
+------------------>*
          |
/00021100/*   17
        *   *
      *       *
 (-/0) *       *
+-----*     Q     *
|     *           *
|       *       *
|         *   *
|           *
|           | (+)
|           |
|           |
|           |
|           |
|           |
|       3   *   18
|         *   *
|       *       *
|     *           *  TRUE
|   * SELCRS .NE.  *-->
|   *  COP(I)     *  |
|     *       *     |
|       *   *       |
|         *         |
|           |FALSE ----
|           |     . 4 .
|           |     . 10 .
|           |      ----
|           |        1
|           |
+---------->|
|           |
|       4   *   19
|         *   *
|       *       *
| (-/0) *       *
+-----*     W     *
|     *           *
|       *       *
|         *   *
|           *
|           | (+)
|           |
|           |
|           |
|           |
|           |
|       5   *   20
|         *   *
|       *       *
|     *           *  TRUE
|   * SELSCH .NE.  *-->
|   * SCHOOL(I) *   |
|     *       *     |
|       *   *       |
|         *         |
|           |FALSE ----
|           |     . 4 .
|           |     . 10 .
|           |      ----
|           |        1
|           |
+---------->|
|           |
|       6   |        21
|   *-------------------*
|   | CRSES = CRSES + 1 |
|   *-------------------*
|           |
|           |
|/00021800/*   22
|         *   *
|       *       *
|     *           *  TRUE
|   * CRSES .LT.   *-->
|   *    176      *  |
|     *       *     |
|       *   *       |
|         *         |
|           |FALSE ----
|           |     . 4 .
|           |     . 10 .
|           |      ----
|           |        1
|           |
|           |
|           |
EXIT MODEL IF NO
ACTIVE COURSES
|           |
|           |
|           |  /
|           -----
|         / 5.01
```

FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

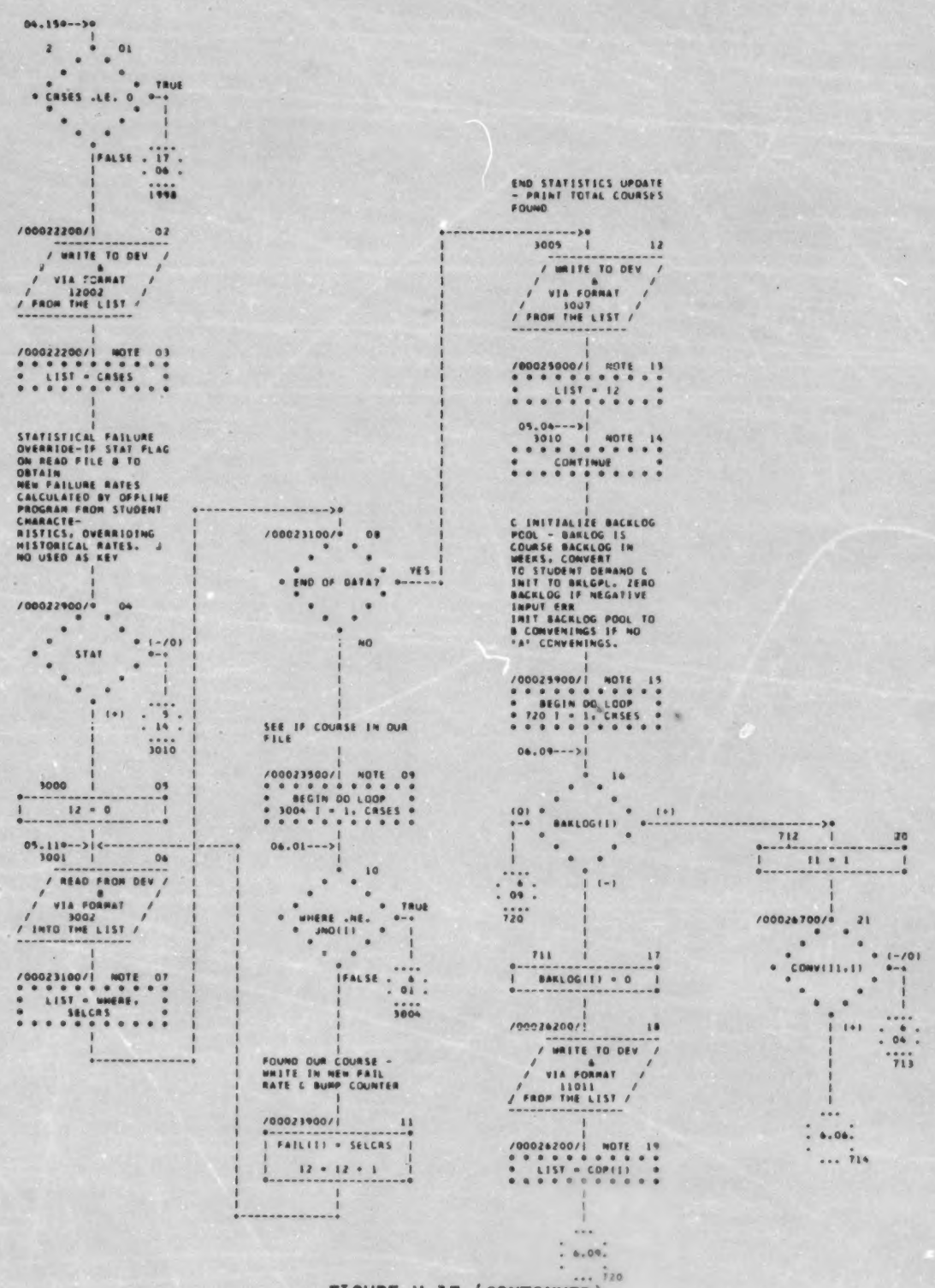


FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

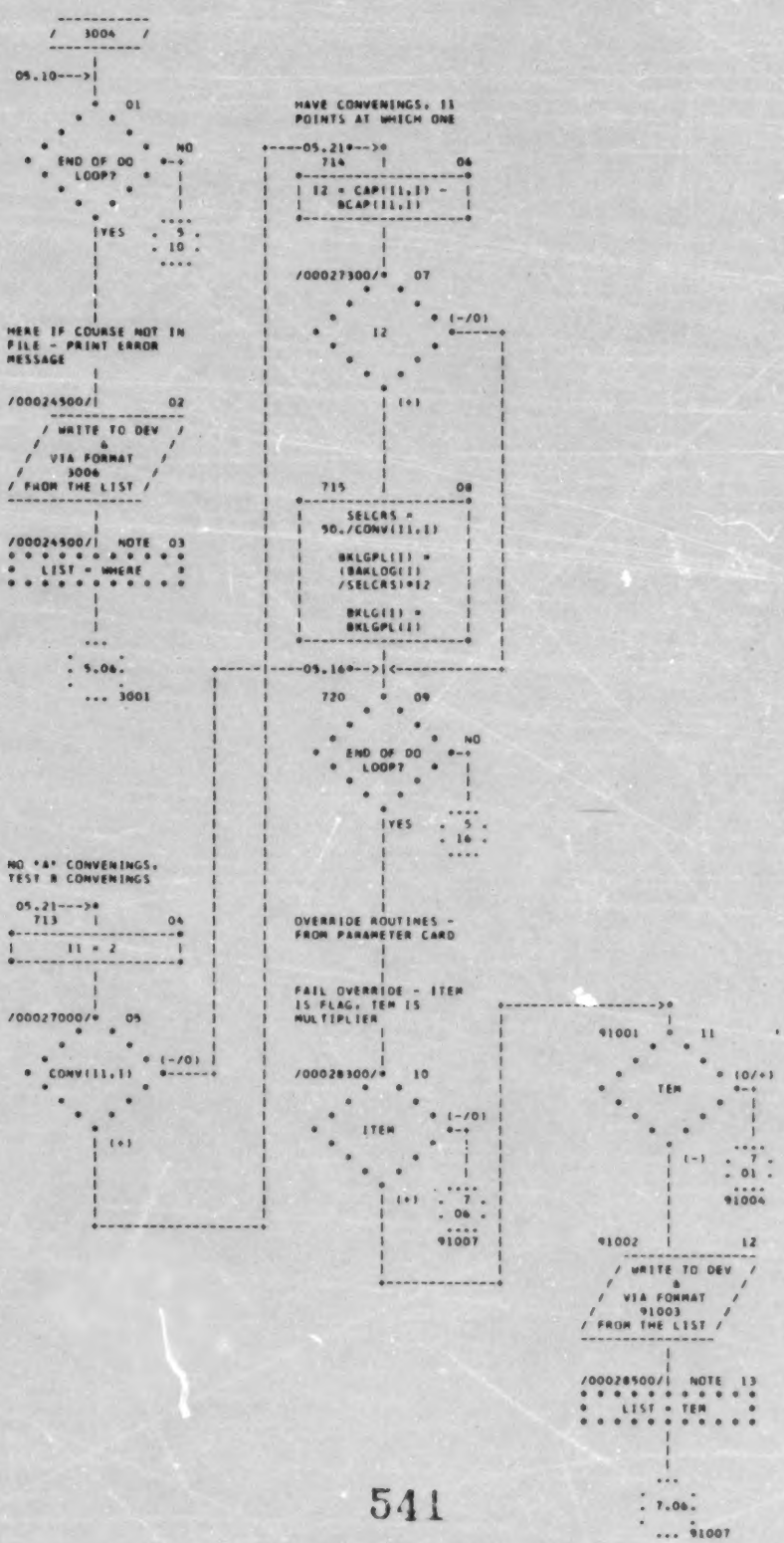


FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

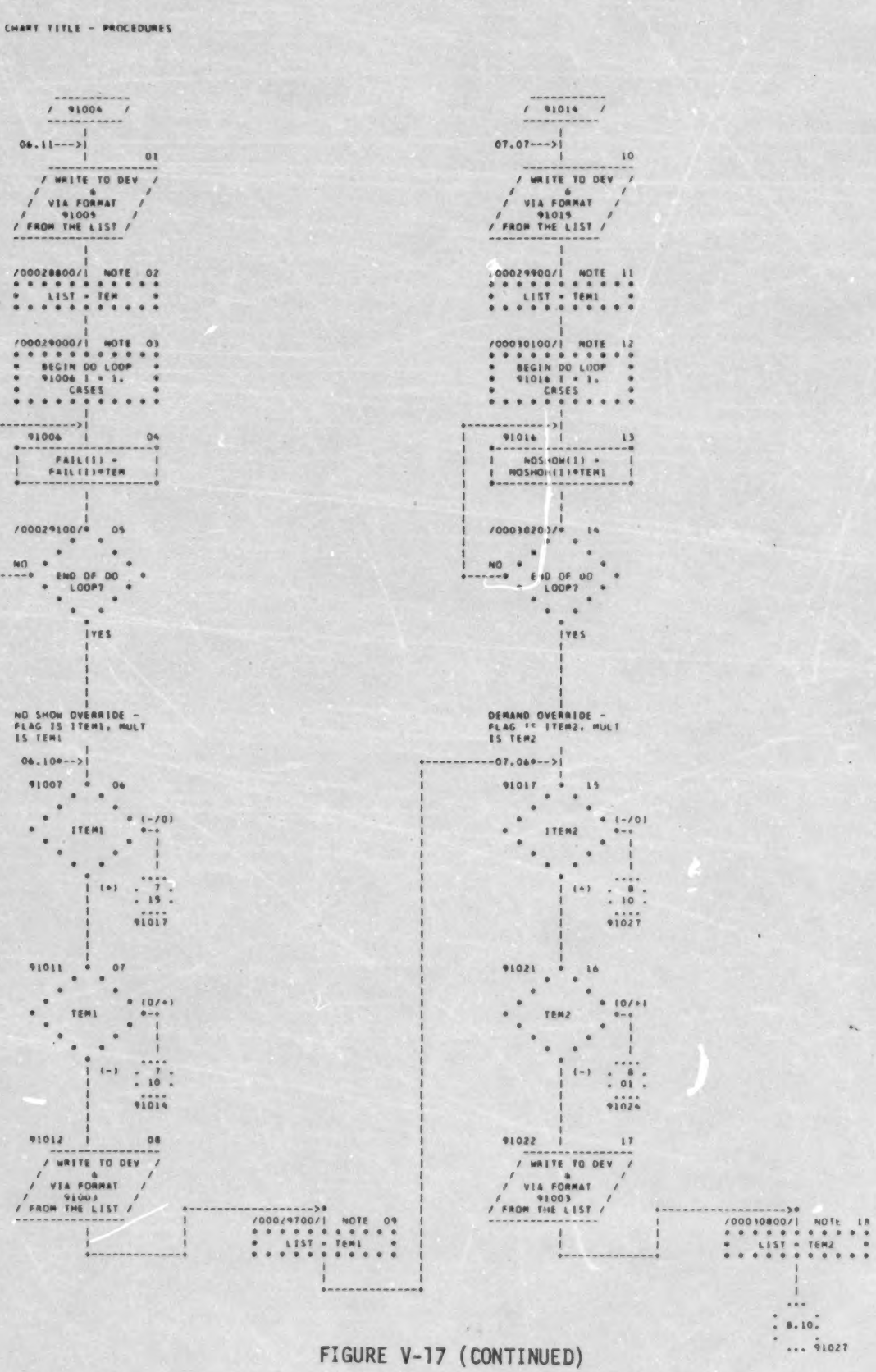


FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

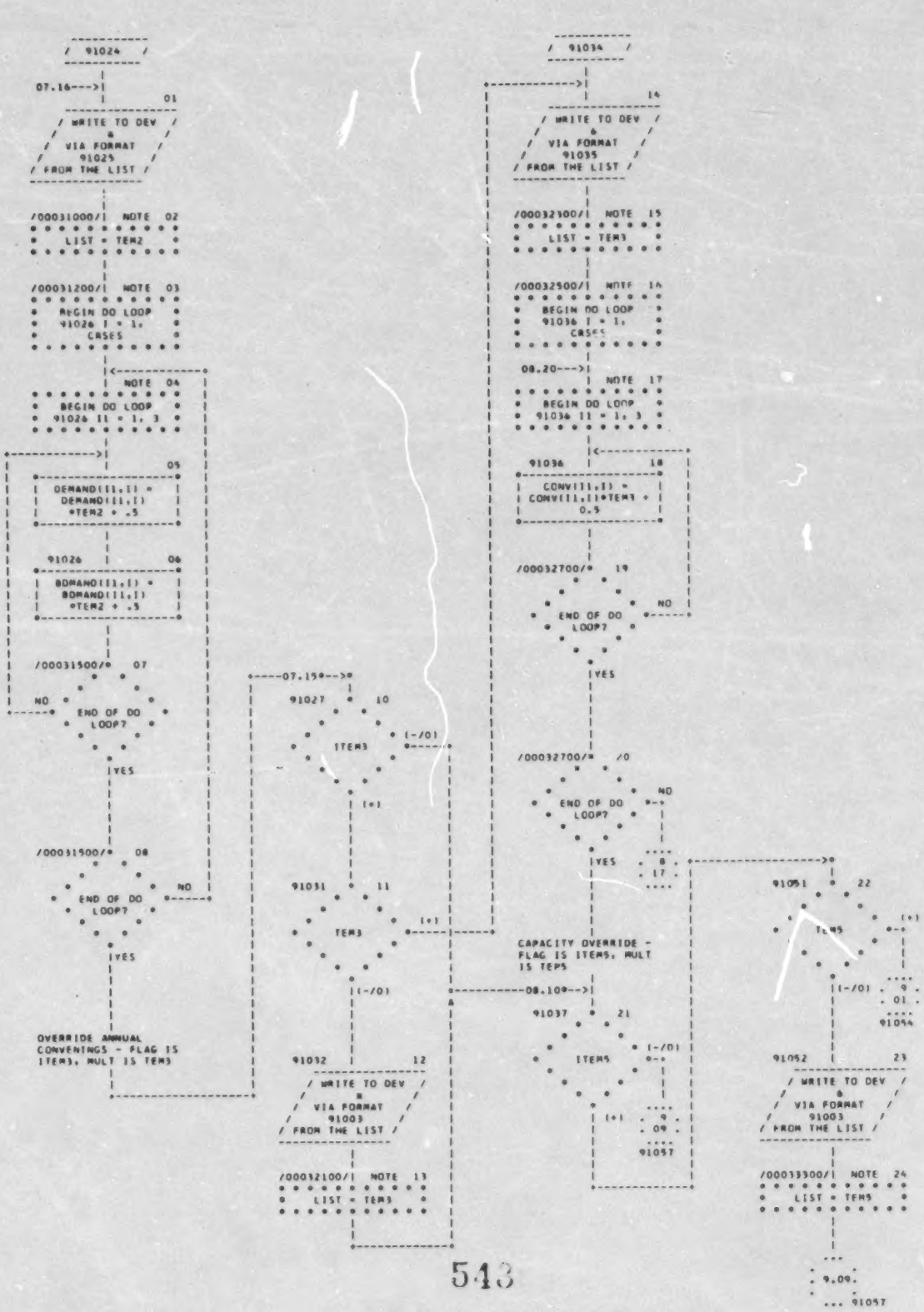


FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

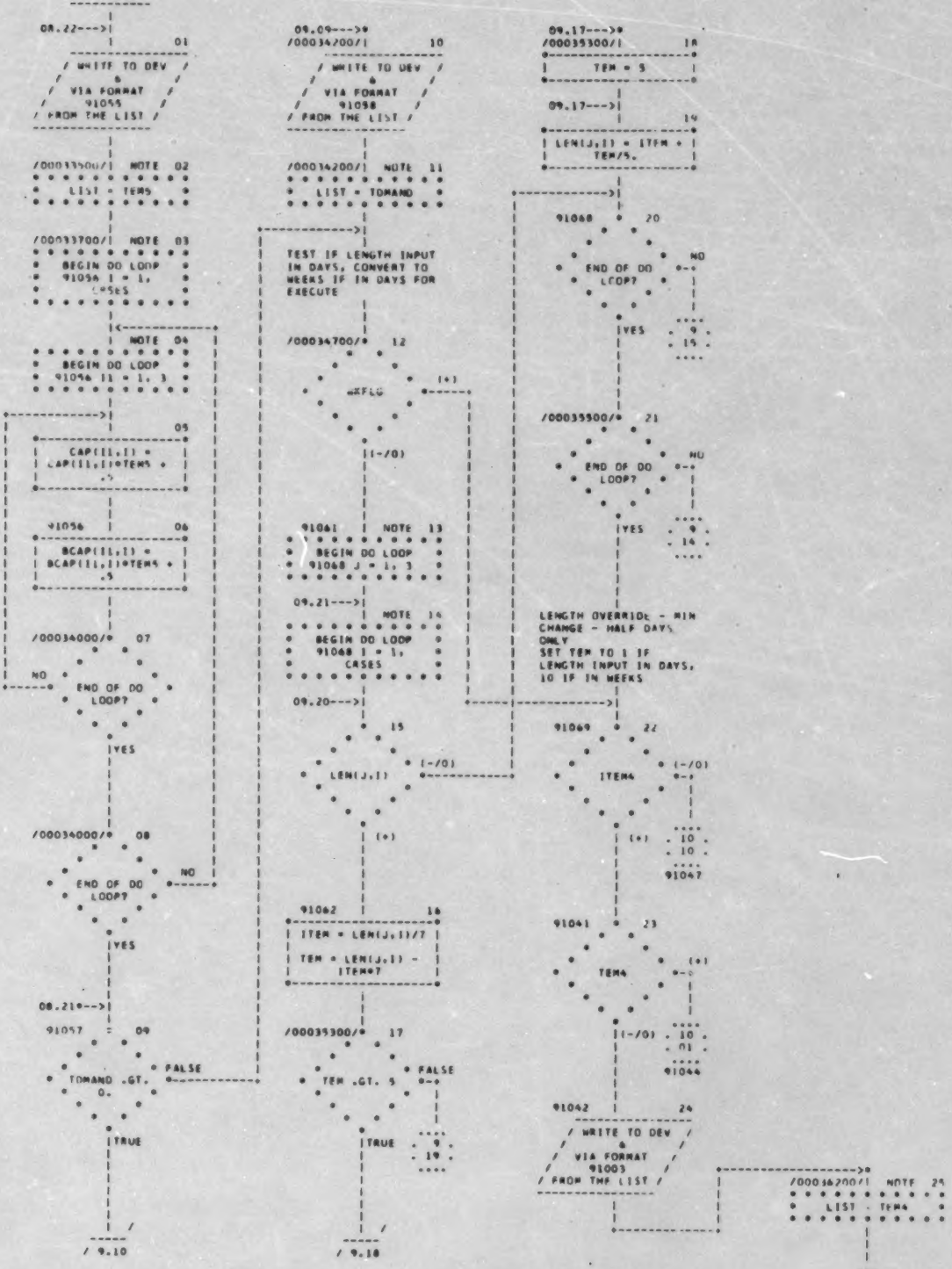
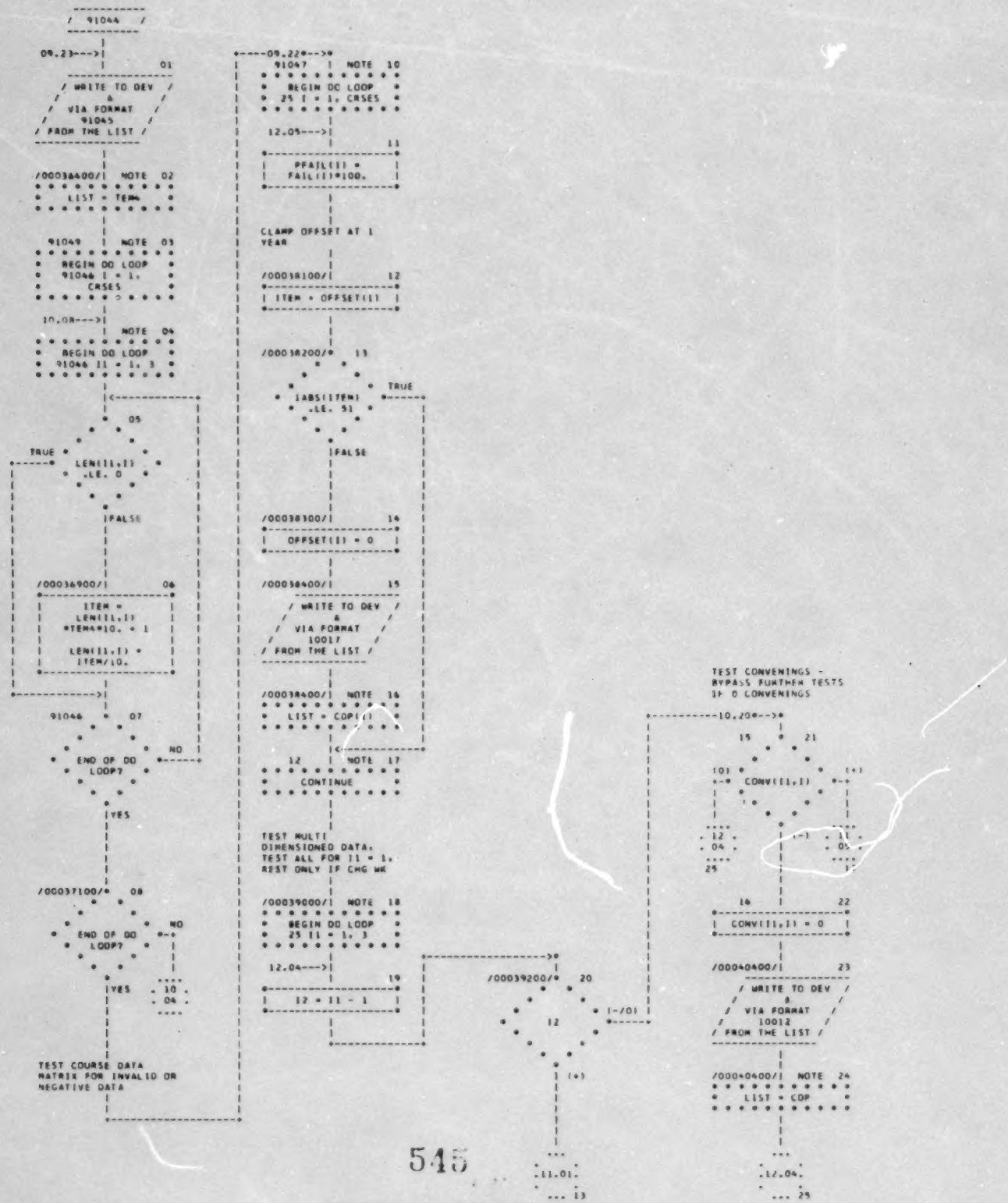


FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES



545

FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

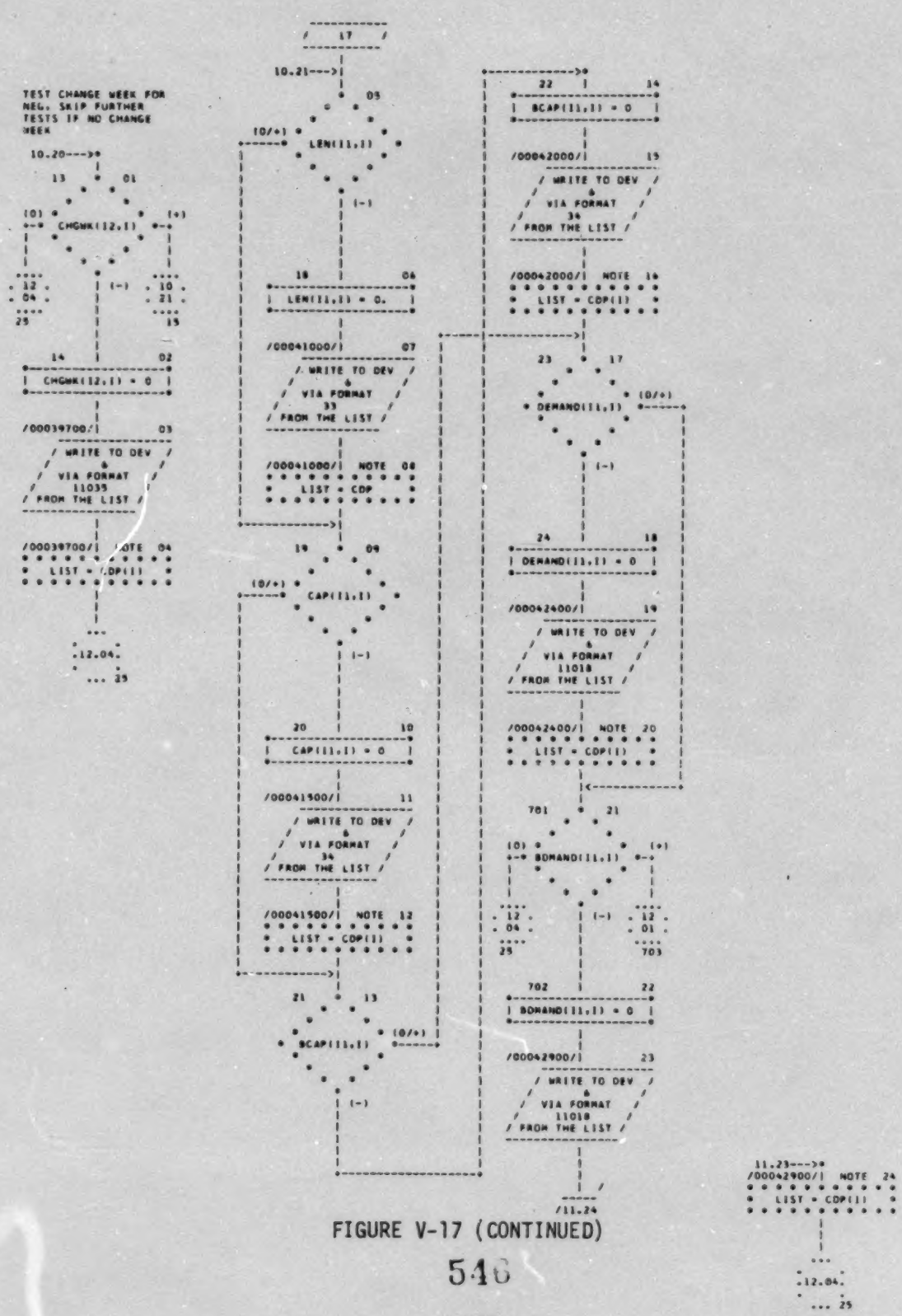
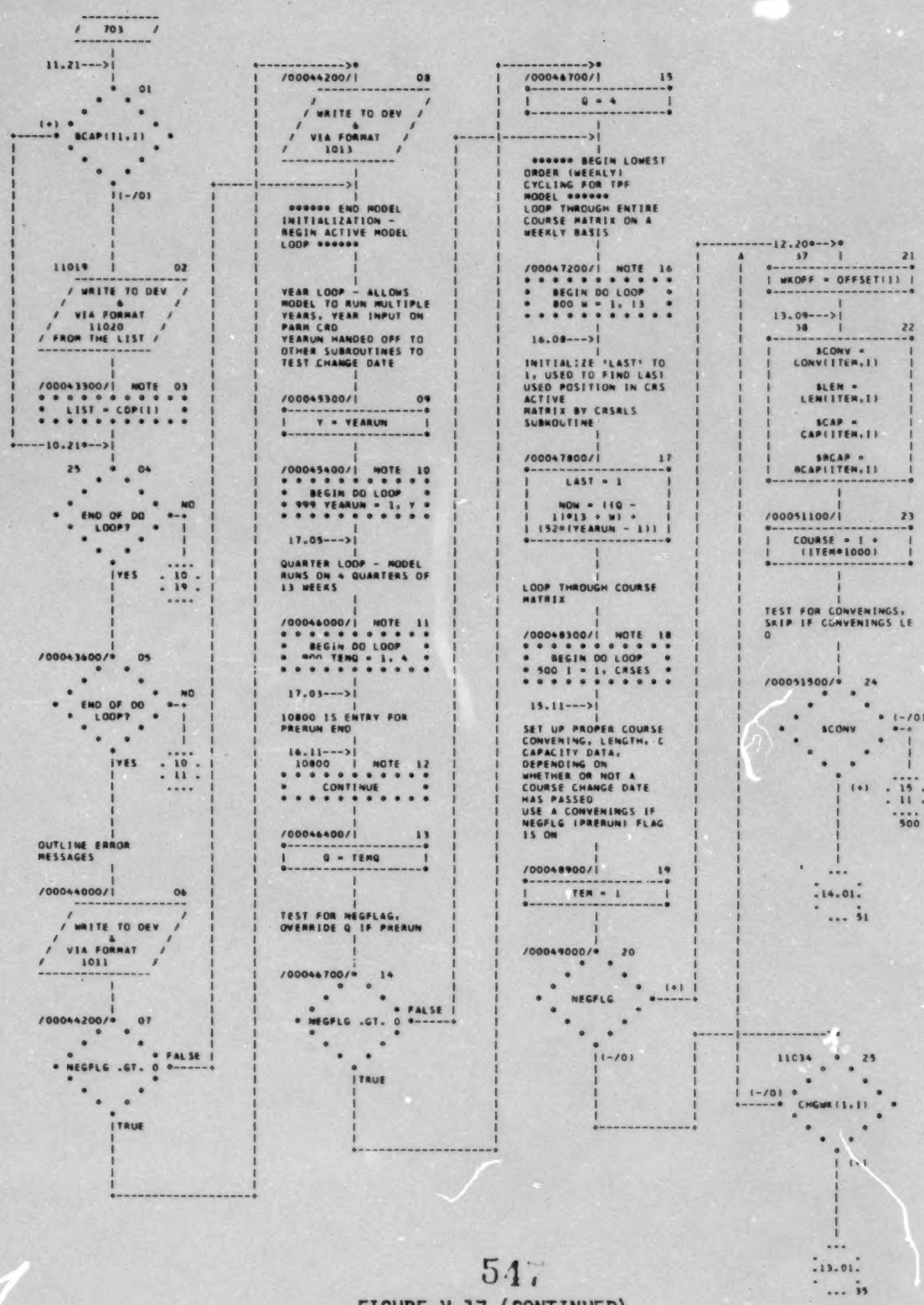


FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES



547

FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

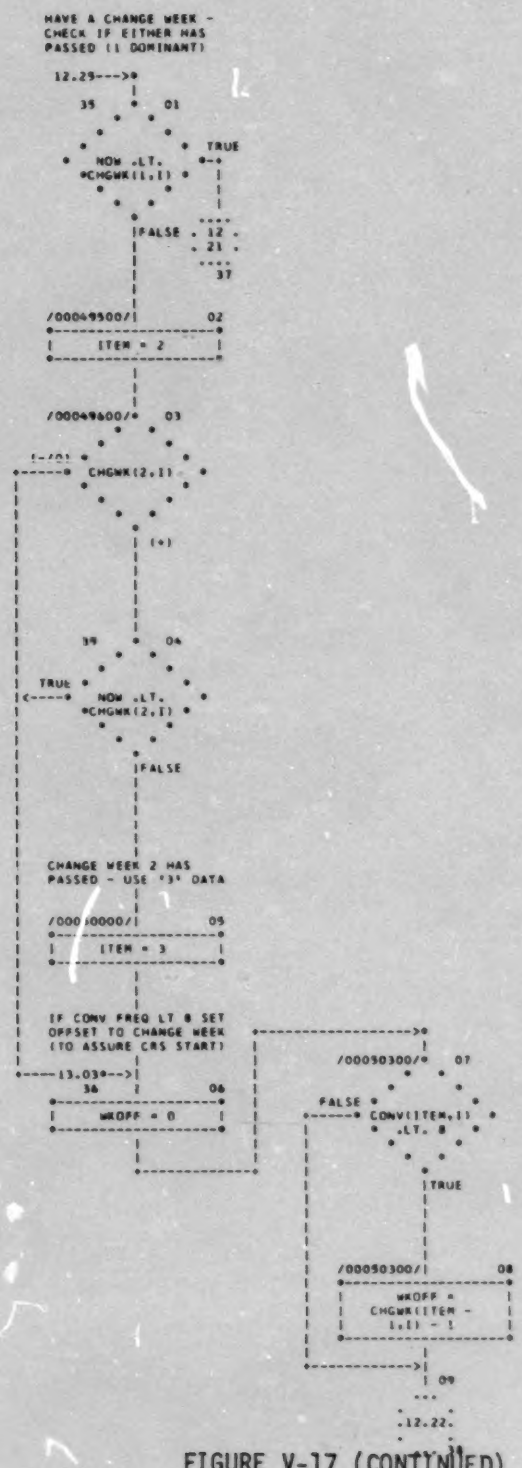


FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

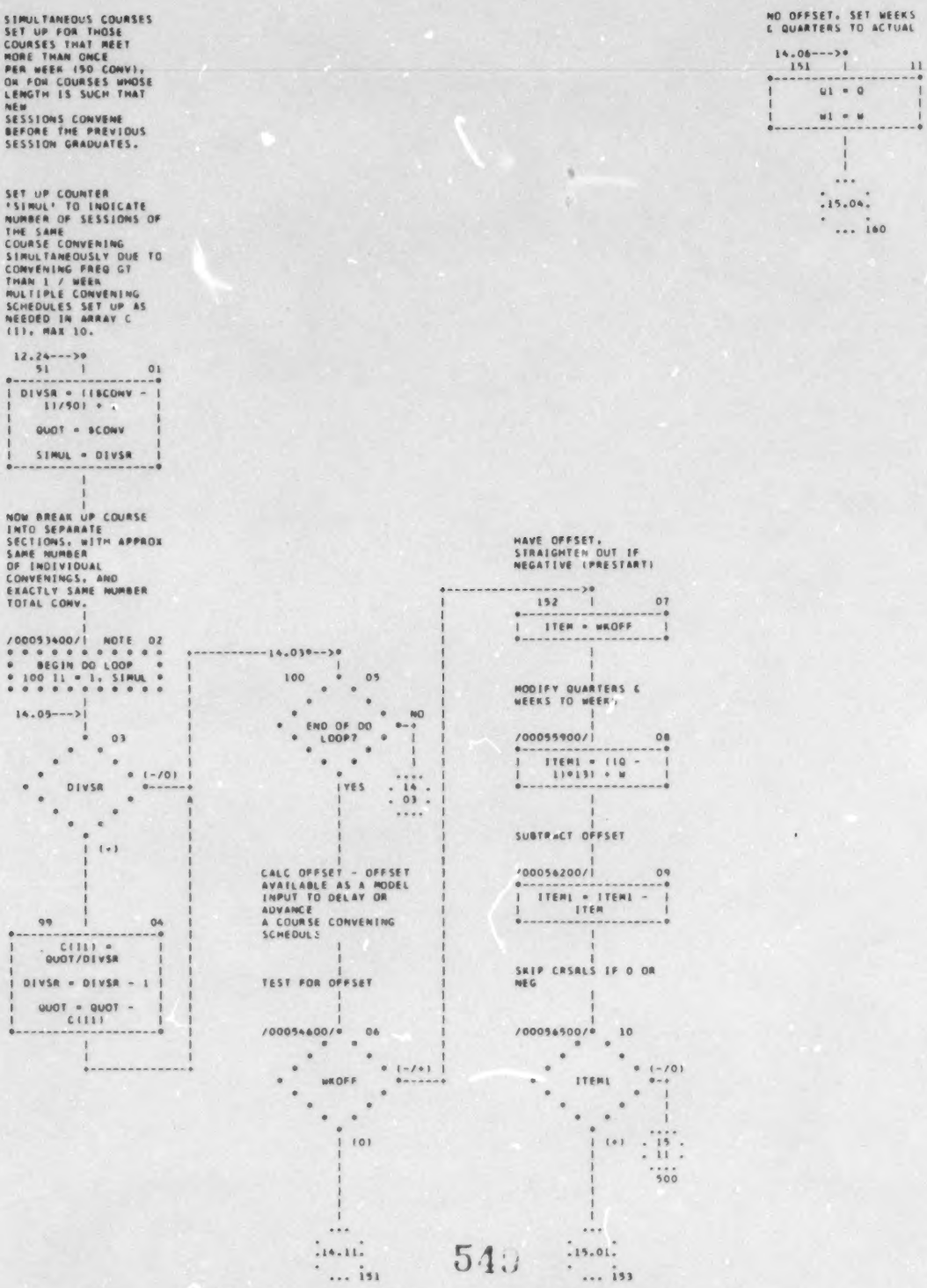


FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

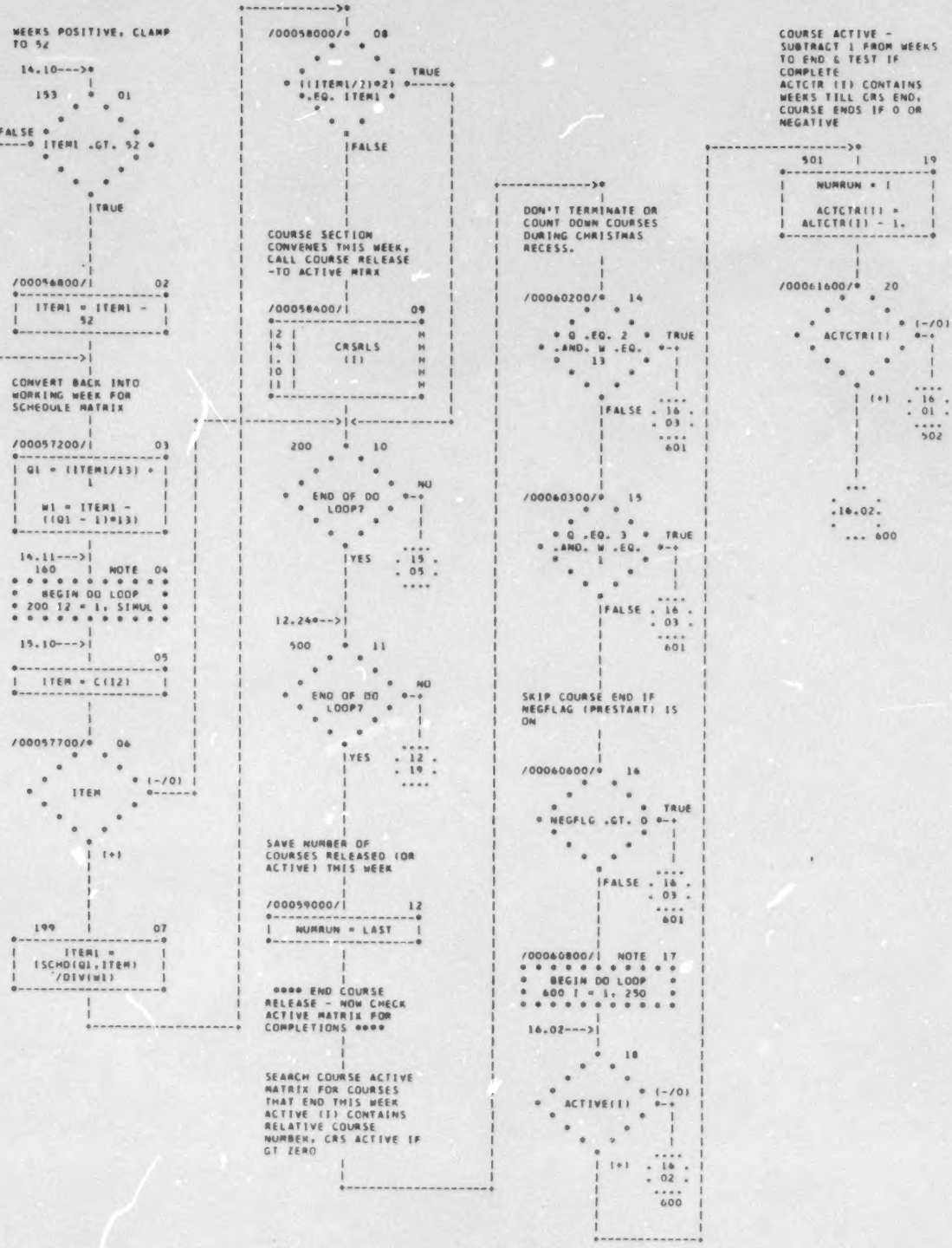


FIGURE V-17 (CONTINUED)

550

CHART TITLE - PROCEDURES

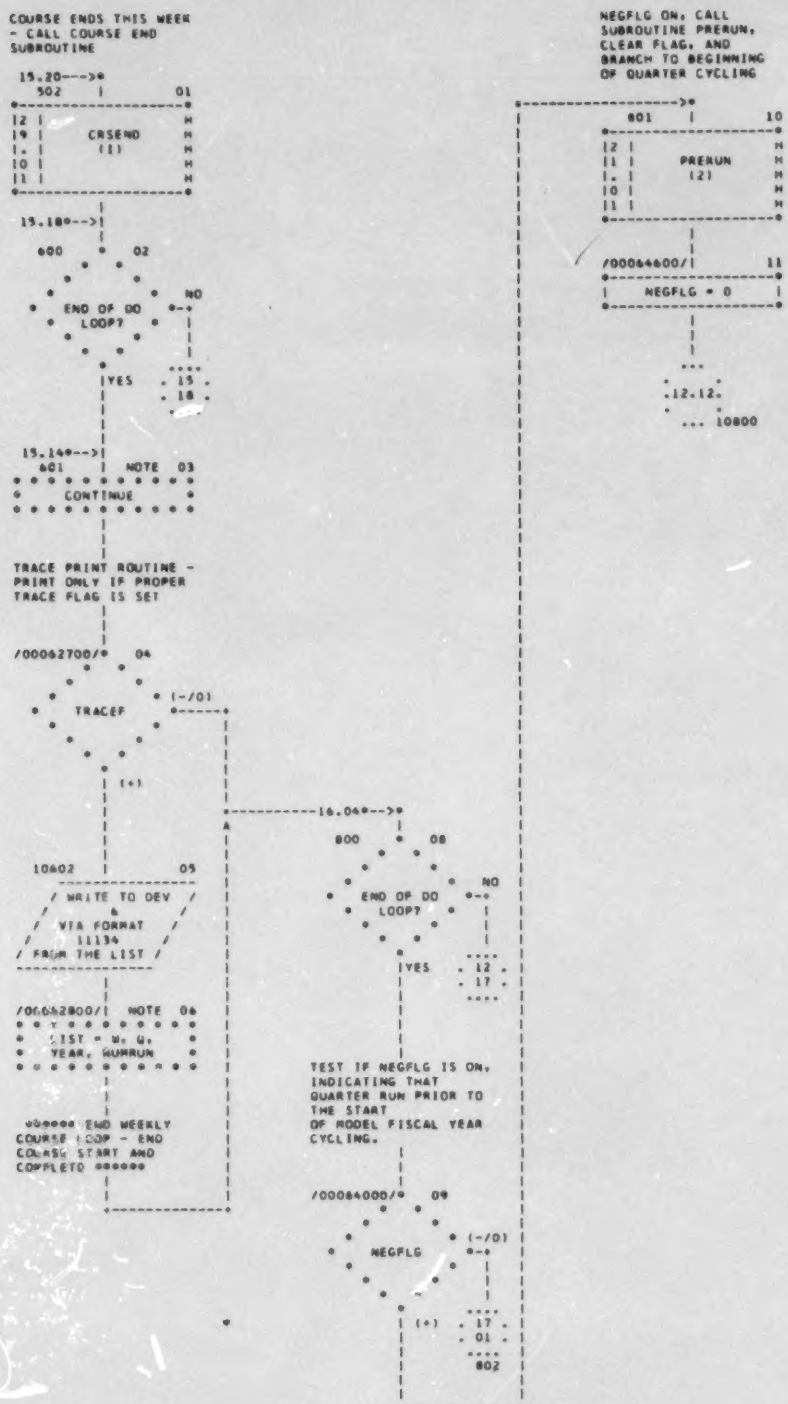


FIGURE V-17 (CONTINUED)

CHART TITLE - PROCEDURES

FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
IMPLICIT INTEGER*2 (A-Z,$)
REAL*4 SELCRS
INTEGER*4 SELSCH
REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
REAL*4 TDMAND, SETBAK
COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
COMMON /DATA/ BKLG (176), TDMAND, SETBAK
INTEGER*4 JNO, PNEC, SCHOOL
REAL*4 PAOB, PFAIL
REAL*8 NAME1, NAME2, CATNO1
COMMON /TEXT/ JNO (176), NAME1 (176), NAME2 (176), CATNO1 (176)
COMMON /TEXT/  CATNO2 (176), PNEC (176), TYPE (176), PAOB (176)
COMMON /TEXT/ PFAIL (176), PRTY (176), SCHOOL (176)
REAL*4 TFAIL, TNADIS
INTEGER*4 TSDAYS, TCAP, TUTIL
COMMON /TOTALS/ TCAP (5, 176),  TUTIL (5, 176),  TSETBK (5, 176)
COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
REAL*4 ACTCTR
COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)
INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN
COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE
COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, ADBCON
COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
DIMENSION C (10), ISCHO (4, 50), DIV (13)
DATA ISCHO/Z1000,3*Z0,Z1000,Z0,Z800,Z0,Z1000,Z100,Z4,Z0,2*Z1000,
Z800,Z1000,Z1004,Z20,Z40,Z200,Z1008,Z100,Z804,Z80,Z810,Z408,Z81,
Z40,Z1020,Z1040,Z810,Z820,Z841,2*Z104,Z410,Z1084,Z420,Z842,Z210,
Z1084,Z444,Z108,2*Z888,Z1110,Z844,Z888,Z889,Z222,Z112,Z444,Z1112,
Z488,Z489,Z244,Z1224,Z1124,Z248,Z1244,Z1224,Z1248,Z892,Z924,Z925,
Z492,Z249,Z492,Z1249,Z4A4,Z492,ZAAA,ZA8A,ZA52,Z252,ZA49,Z1294,
Z14A4,Z94A,ZA52,ZA95,Z54A,Z4A5,Z952,ZA95,Z554,Z555,Z52A,2*ZAAA,
Z1AA,2*ZAAA,Z1554,Z555,ZAAA,Z1555,ZAAA,Z555,ZAAA,Z1AAA,Z1554,ZAAA,
Z1555,Z1AAA,Z156A,ZA85,ZAAA,Z156A,Z1AAA,ZAAA,Z1AD5,Z156A,Z1AR4,
Z85A,Z16AD,ZD68,Z85A,Z685,Z15AD,Z1585,Z15AC,ZDAD,Z1586,ZD86,Z185A,
Z86D,Z1586,Z16DA,Z186C,ZD86,Z186D,ZDDB,ZD86,Z76D,Z16DB,ZDD8,ZED4,
ZD87,ZDBB,ZEED,Z1876,Z876,Z1DBB,Z1776,Z1DDC,ZEED,Z1888,Z1777,ZEEE,
Z78B,Z1777,ZF78,Z188A,ZEF7,Z1777,ZF7B,Z18DE,Z78D,Z1DEF,Z17RE,
Z1EFA,ZEFB,Z18EF,ZFDF,ZFBE,Z7EF,Z17DF,Z17EF,Z18F6,ZF7E,Z1FBF,ZFF7,
Z1EFE,ZF7B,Z1F7F,ZFFB,Z1FDE,ZFBF,Z1DFF,ZFFF,ZFFE,Z7FF,2*ZFFF,
Z1EFE,ZFFB,Z1FFF,ZFFF,Z1FFE,Z7FF,Z1FFF,ZFFF,Z1FFE,ZFFF,2*Z1FFF,
```

FIGURE V-17 (CONTINUED)

553

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
        Z1FFE,ZFFF,Z1FFF/
        DATA DIV/4096,2048,1024,512,256,128,64,32,16,8,4,2,1/
1000    FORMAT(I2, 3I4, 1X, 7 (I1, F4.3), 5X, I1, F4.3, 2I1, A4),10I1)
1001    FORMAT ('1BEGIN TRAINING PROCESS FLOW MODEL EXECUTION - CHARACTERI
        STICS FOR THIS RUN ARE:')
1002    FORMAT (/ ' FIRST FISCAL YEAR OF MODEL RUN', I5, 4X, 'YEARS PROJEC
        TED', I3)
1006    FORMAT (/' AOB CONSTANT IS', I4, '  WEEKEND DAYS USED IN STUDENT D
        AYS AND AOB CALC')
1007    FORMAT (/' AOB CONSTANT IS', I4, '  ONLY WORKING DAYS USED IN STUD
        ENT DAYS AND AOB CALC')
1008    FORMAT (/ ' COURSE DATA INPUT DEVICE - READER:    STATISTICS INPUT
         DEVICE - READER')
1009    FORMAT(/' ALL INPUTS THIS RUN FROM UNLOADED DATA BASE')
1010    FORMAT(/' TRACE FLAGS THIS RUN:  PRERUN', I2, ': CRS REL', I2, ':
        CRS END', I2, ': YEAR FORMAT', I2 /)
1012    FORMAT ( ' INIT FLAG IS SET - MODEL WILL INIT COURSES IN SESSION A
        T 1ST QUARTER START' /)
11013   FORMAT ( ' STAT FLAG IS SET-STATISTICAL FAIL DATA WILL OVERRIDE HI
        STORICAL RATES' /)
1011    FORMAT ( ' ------------------------------------------------------
        ---------- -------------')
12014   FORMAT (' SELECT OPTION- SELECT KEY IS ', A4)
10      FORMAT (A4, A8, A2, 2A8, I3, F3.1, 2I3, 2I5, I2, 6F4.3, I4, A4,2X,
         /4X,
         2 (I3, F3.1, 2I3, 2I5, I3), 2F4.1, 2A2, 2A4)
12002   FORMAT (' COURSES ACTIVE THIS RUN', I5)
3002    FORMAT (A4, F4.3)
3006    FORMAT (' STATISTICS OVERRIDE-COURSE J NO ', A4, ' NOT IN FILE')
3007    FORMAT ('STATISTICS UPDATE ROUTINE - ', I3, ' COURSES UPDATED')
11011   FORMAT (' NEG BACKLOG COURSE ', A4, '  ZEROED THIS RUN')
91003   FORMAT (' NEGATIVE OVERRIDE - REQUEST IGNORED. VALUE =', F6.3)
91005   FORMAT (' FAIL OVERRIDE- RATES MULT BY', F6.3, ' THIS RUN ONLY')
91015   FORMAT (' NO SHOW OVERRIDE - RATES MULT BY',F6.3,' THIS RUN ONLY')
91025   FORMAT (' DEMAND OVERRIDE-DEMANDS MULT BY',F6.3,' THIS RUN ONLY')
91035   FORMAT (' CONVENINGS OVERRIDE-FREQ MULT BY',F6.3,' THIS RUN ONLY')
91055   FORMAT (' CAPACITY OVERRIDE-CAP MULT BY',F6.3,' THIS RUN ONLY')
91058   FORMAT (' TREND OPTION- DEMANDS MULT BY', F6.3, ' EACH YEAR')
91045   FORMAT (' LENGTH OVERRIDE-LEN MULT BY',F6.3,' THIS RUN ONLY')
10017   FORMAT (' OFFSET GT 1 YEAR FOR COURSE ', A4, '  ZEROED THIS RUN')
11035   FORMAT (' NEGATIVE CHANGE WEEK COURSE ',A4,' ZEROED THIS RUN')
10012   FORMAT (' NEGATIVE CONVENINGS FOR COURSE ', A4,' ZEROED THIS RUN')
33      FORMAT (' NEG LENGTH COURSE ', A4, '  ZEROED THIS RUN')
34      FORMAT (' NEGATIVE CAPACITY COURSE ', A4, '  ZEROED THIS RUN')
11018   FORMAT (' NEGATIVE DEMAND FOR COURSE ', A4, '  ZEROED THIS RUN')
11020   FORMAT (' WARNING-BUPERS DEMAND WITH NO BUPERS SEATS COURSE ', A4)
1013    FORMAT(/ ' PRERUN FLAG SET- 4TH QTR WILL BE RUN TO INIT CRSES IN S
        ESSION AT MODEL START',//)
```

FIGURE V-17 (CONTINUED)

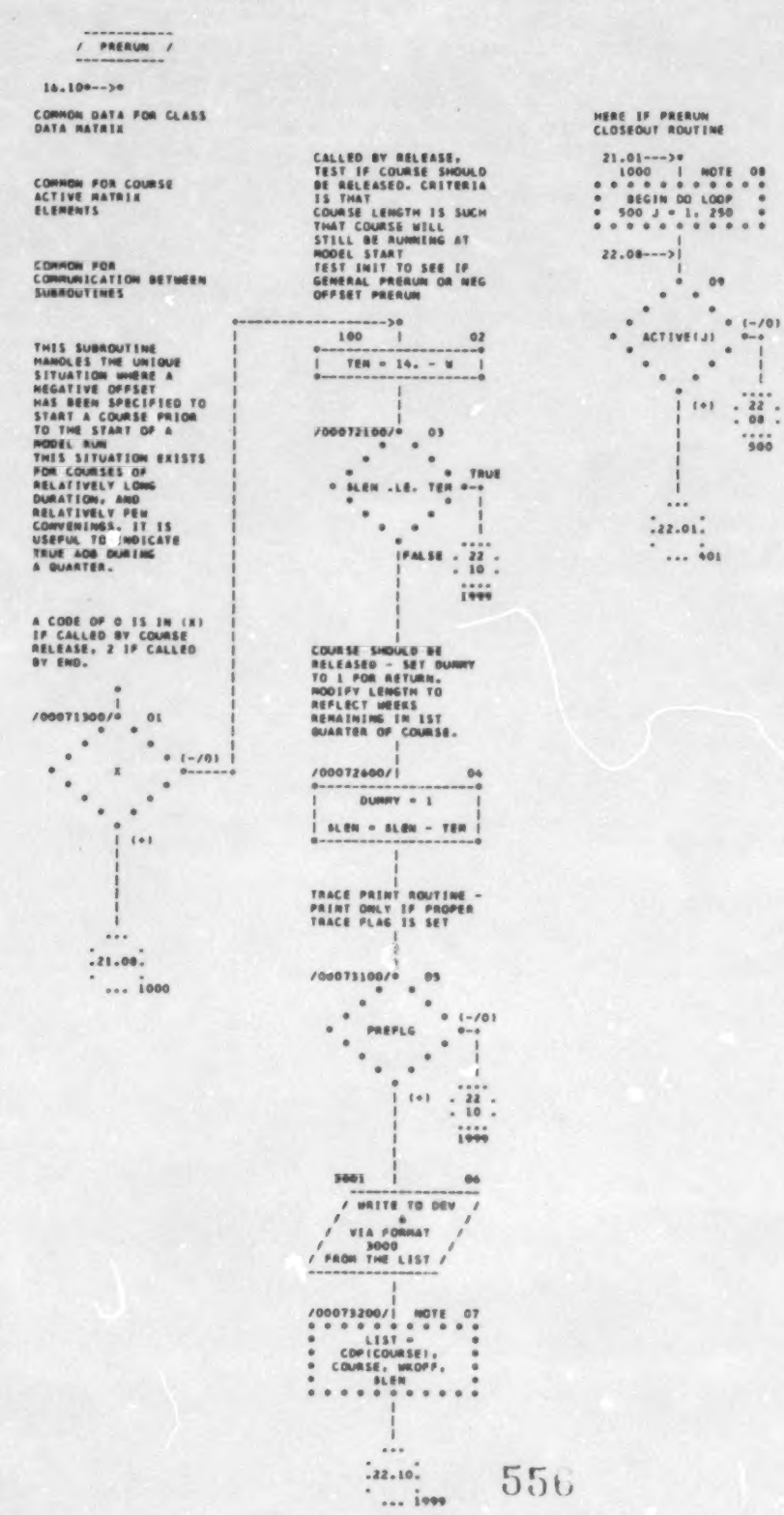
554

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
11134   FORMAT (' TRACE- END OF WEEK    WEEK =', I3, '1   QUARTER =', I2,
        '1   YEAR =', I5, 9X, 'COURSES ACTIVE THIS WEEK =', I4,/)
9000    FORMAT ('1END OF TPF OUTPUT', I4, ' COURSES PROCESSED', / '/'/'*')
```

FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE PRERUN(X)



556

FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE PRERUN(K)

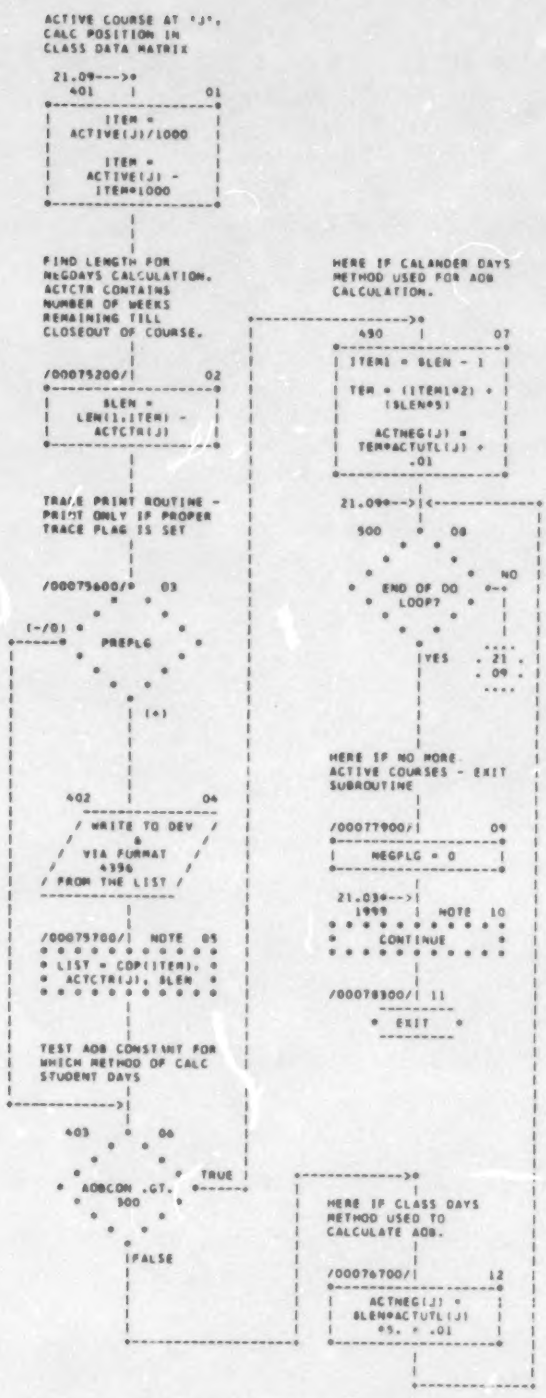


FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      IMPLICIT INTEGER*2 (A-Z,$)
      REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
      REAL*4 TDMAND, SETBAK
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK
      REAL*4 ACTCTR
      COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN
      COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE
      COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
      COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
      COMMON YEAR, CRSES, Q, W, WROFF, INIT, WHERE, NEGFLG, AOBCON
      COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
3000  FORMAT (' PRERUN CALLED - CDP = ', A4, '  POSITION =', I5,
       '  OFFSET =', I4, '  LENGTH IN 1ST QTR =', F4.1)
4356  FORMAT (' PRESTART CLOSEOUT  CDP =', A4, 3X, 'ACTCTR = ', F5.1,
       '   NEGLEN = ', F5.1)
```

FIGURE V-17 (CONTINUED)

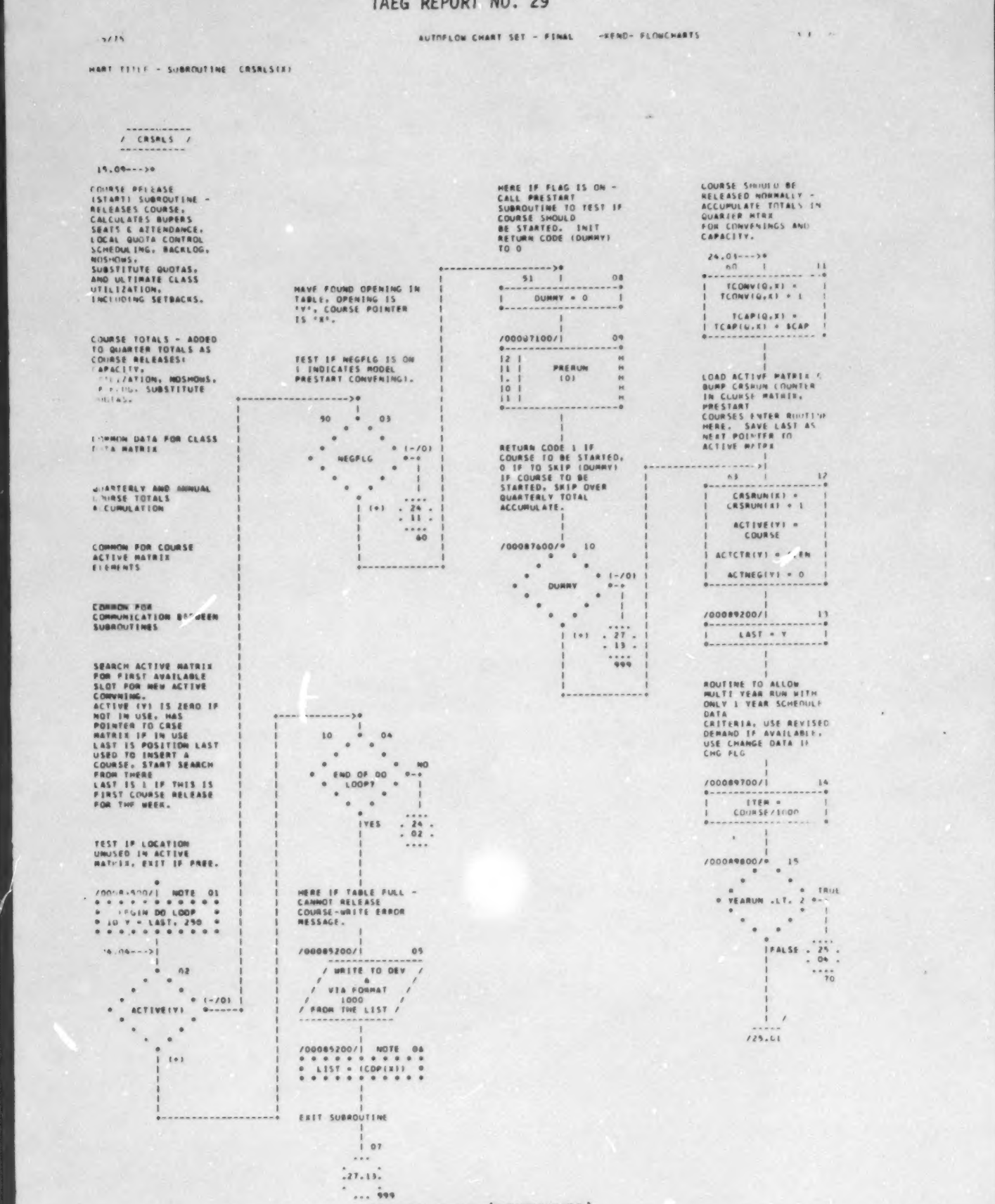


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE CRSRLS(X)

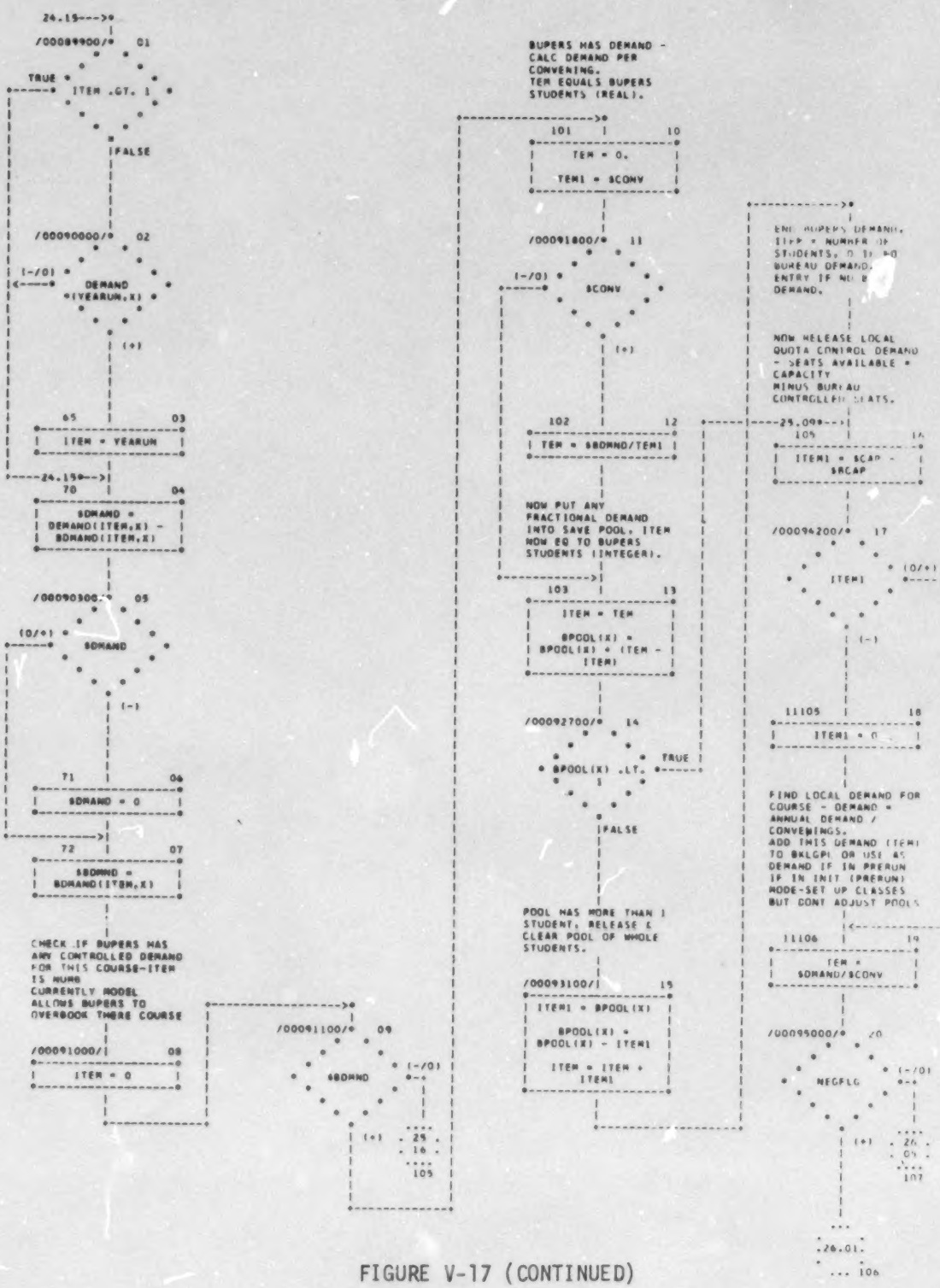


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE CASALS(X)

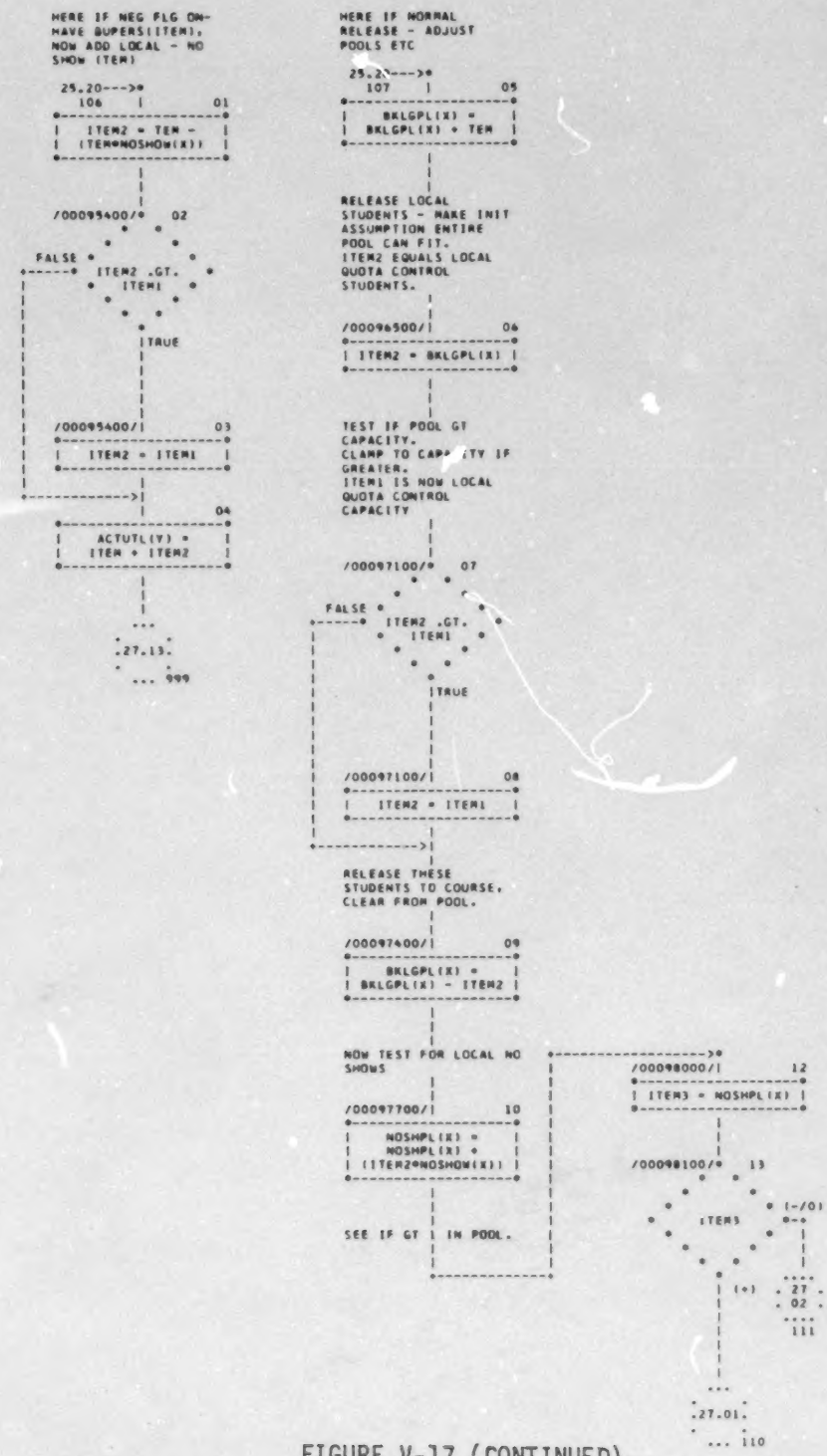


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE CRSRLS(X)

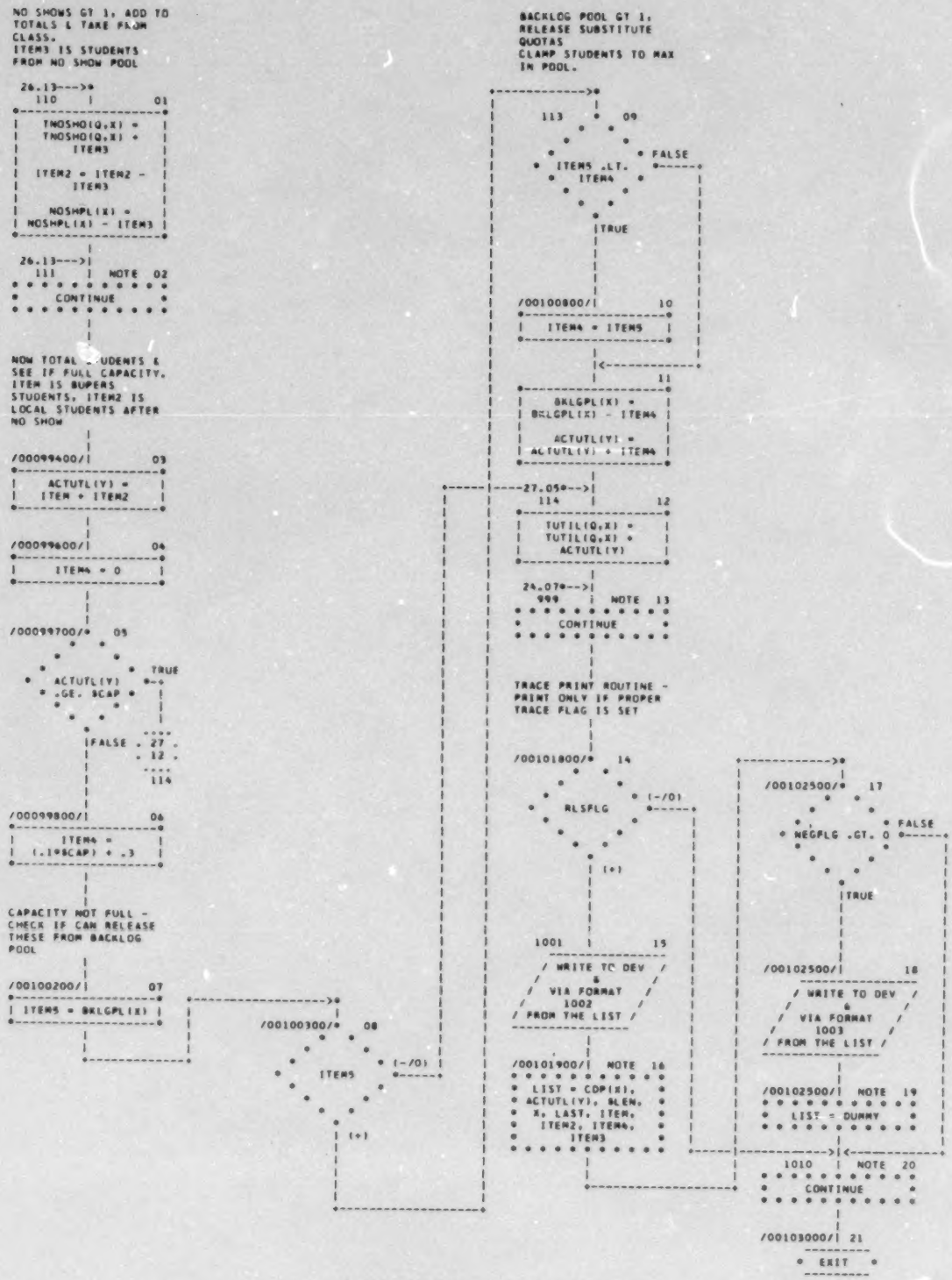


FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      IMPLICIT INTEGER*2 (A-Z,$)
      REAL*4 $DMAND, $BDM'D
      REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
      REAL*4 TDMAND, SETBAK
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (2,176), LEN (3,176)
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK
      REAL*4 TFAIL, TNADIS
      INTEGER*4 TSDAYS, TCAP, TUTIL
      COMMON /TOTALS/ TCAP (5, 176),  TUTIL (5, 176),  TSETBK (5, 176)
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
      COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
      REAL*4 ACTCTR
      COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM', ITEM4, ITEM5, WHERE
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN
      COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE
      COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
      COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
      COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON
      COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
1000  FORMAT (1H0, 'COURSE ACTIVE TABLE FULL - COURSE NOT RELEASED',
       '   COURSE NUMBER IS', A6)
1002  FORMAT (' RELEASE- CDP= ', A4, ';  STUDENTS= ', I3, ';  LENGTH='
      ,F5.1,';  CRS SEQ=',I4,';  RUN POSITN=', I4, ';  BUPRS=', I3, ';
      LOCAL=', I3, ';  SUBQTA=', I2,';  NOSHO=', I3)
1003  FORMAT ('+', T132, 'P', I1)
```

FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE CRSEND(X)

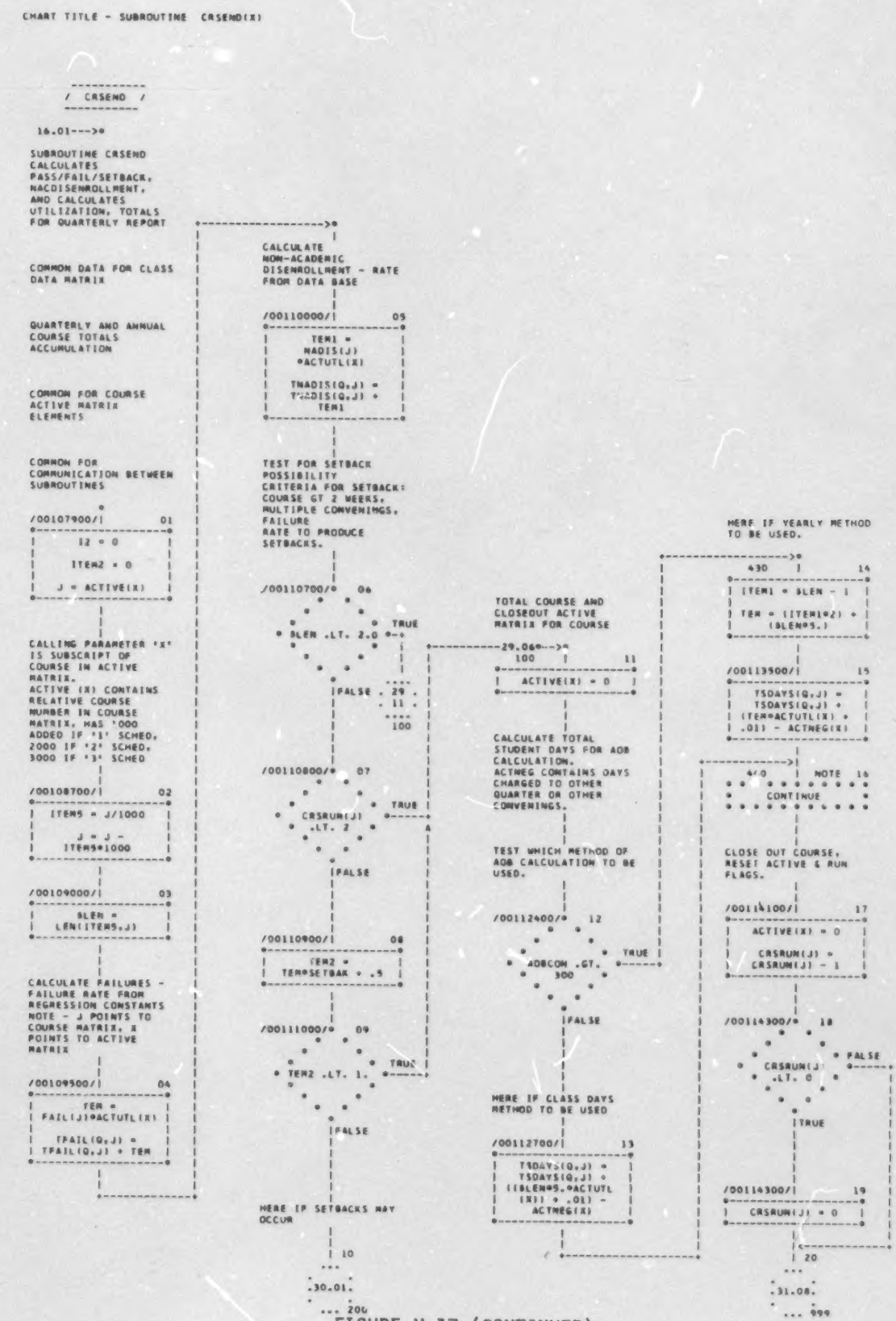


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE CRSEND(X)

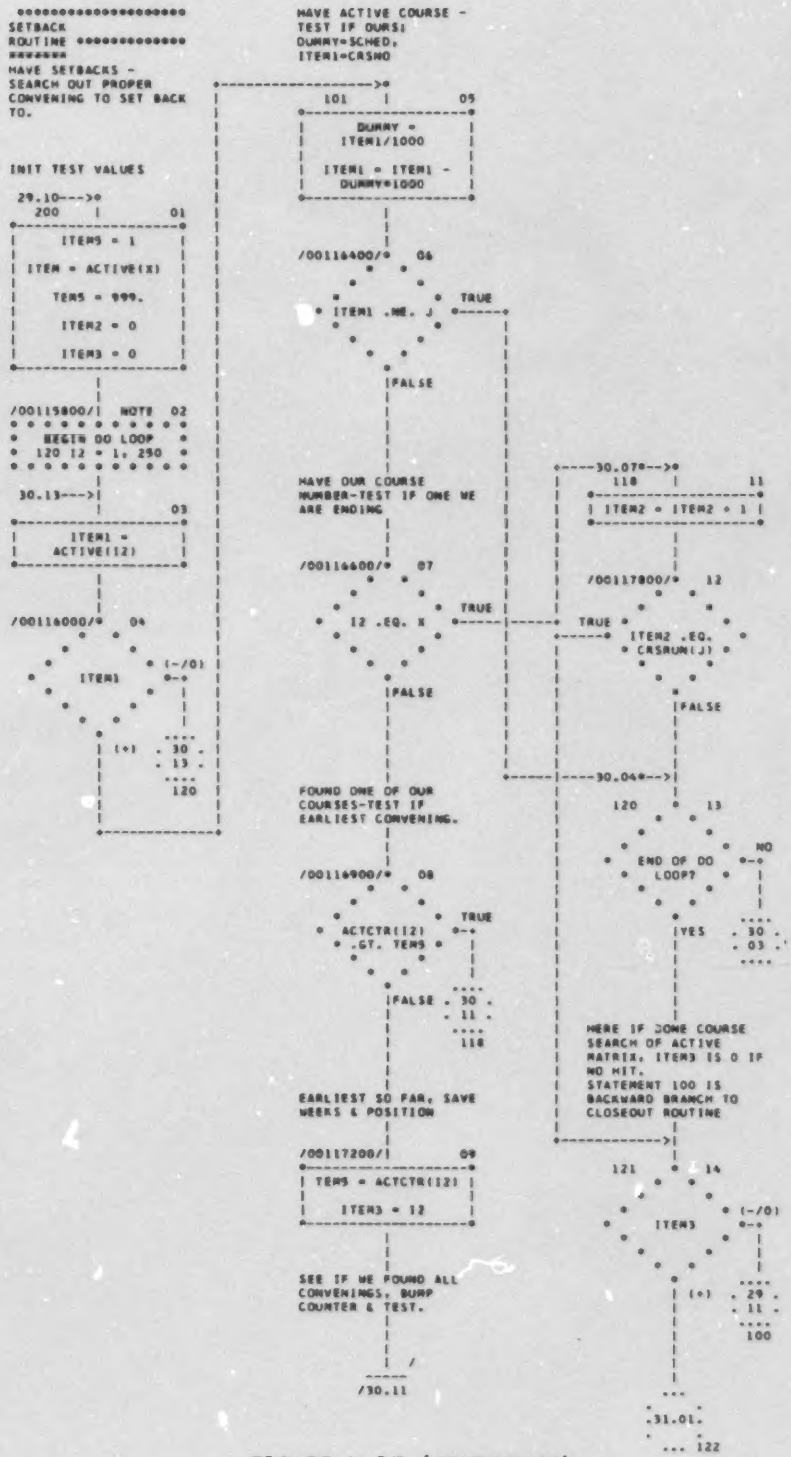


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE CRSEND(X)

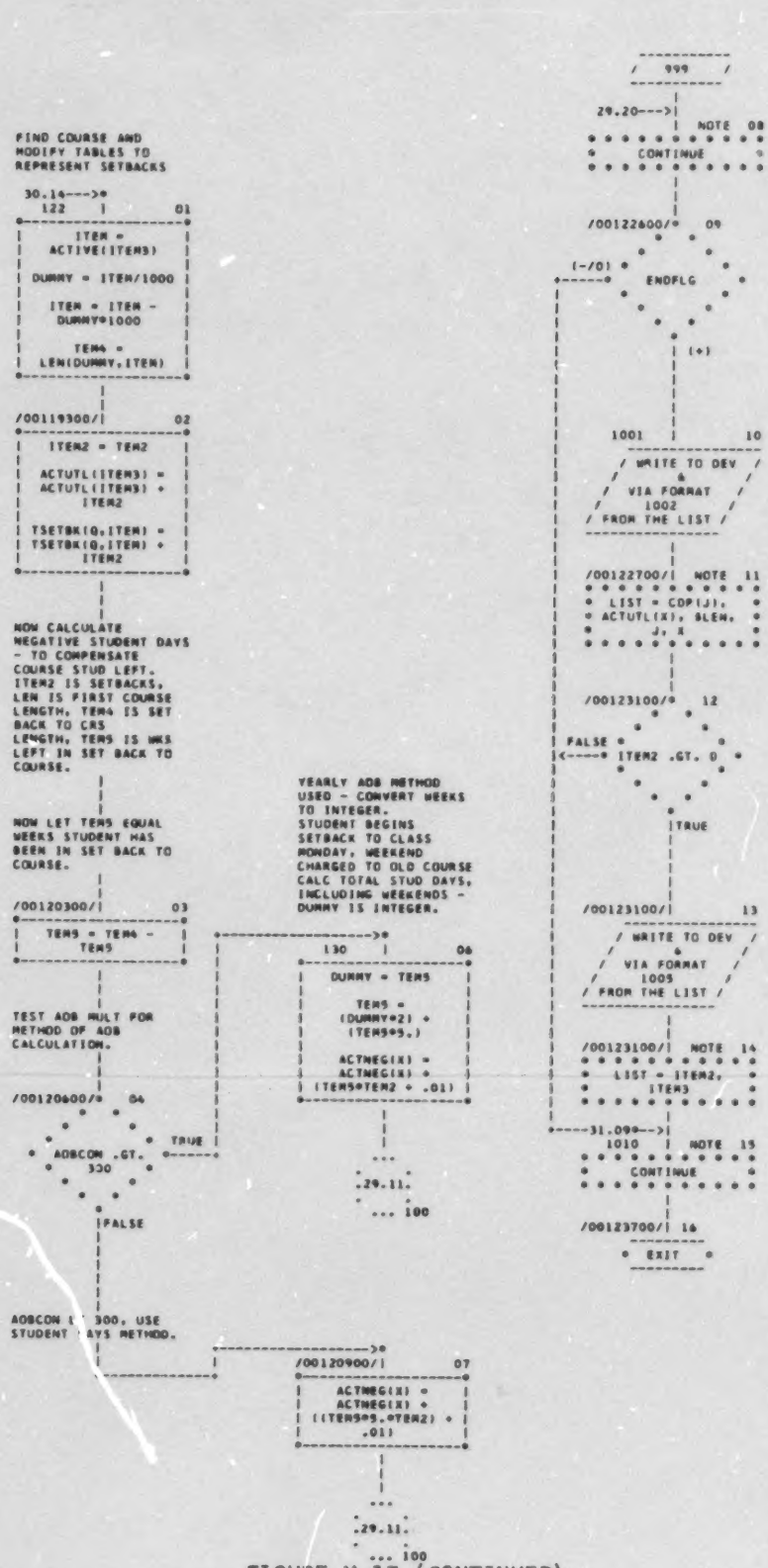


FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      IMPLICIT INTEGER*2 (A-Z,$)
      REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
      REAL*4 TDMAND, SETBAK
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK
      REAL*4 TFAIL, TNADIS
      INTEGER*4 TSDAYS, TCAP, TUTIL
      COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176)
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
      COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
      REAL*4 ACTCTR
      COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN
      COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE
      COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
      COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
      COMMON YEAR, CASES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON
      COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
1002  FORMAT (' CRS END- CDP= ', A4, ';  STUDENTS= ', I3, ';  LENGTH='
      ,F5.1,';  CRS SEQ=',I4,';  RUN POSITN=', I
1005  FORMAT ('+', T105, I2, ' STDNTS SET TO CRS ', I4)
```

FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE QRTEND(IX)

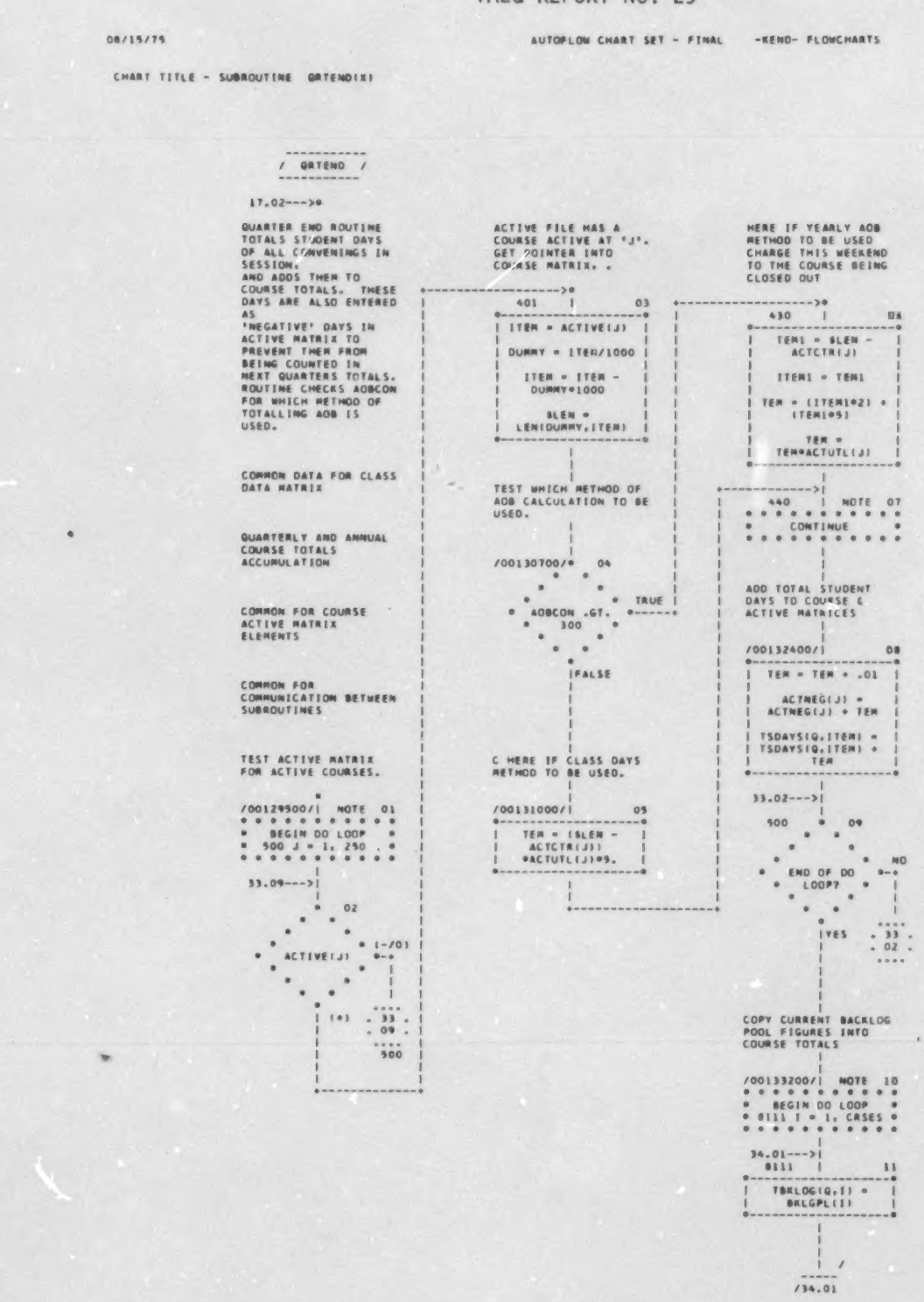


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE QRTEND(X)

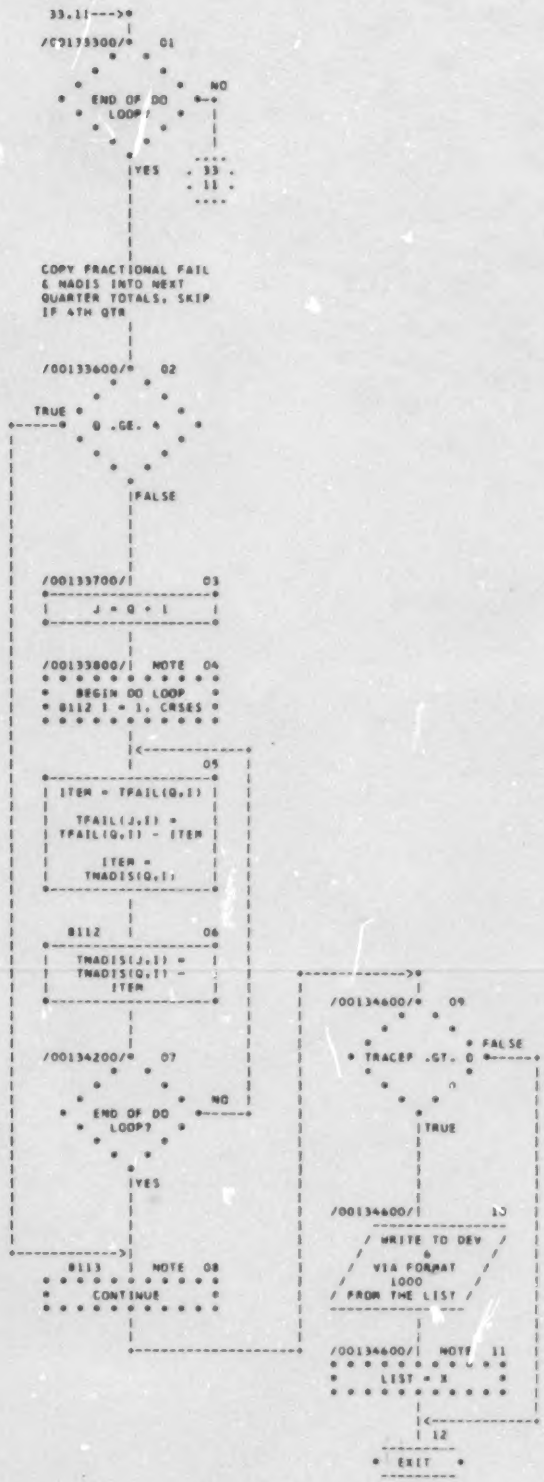


FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      IMPLICIT INTEGER*2 (A-Z,$)
      REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
      REAL*4 TDMAND, SETBAK
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK
      REAL*4 TFAIL, TNADIS
      INTEGER*4 TSDAYS, TCAP, TUTIL
      COMMON /TOTALS/ TCAP (5, 176),  TUTIL (5, 176),  TSETBK (5, 176)
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
      COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
      REAL*4 ACTCTR
      COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN
      COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE
      COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
      COMMON ITEM, ITEM1, ITEA2, ITEM3, ITEM4, ITEM5
      COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON
      COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
1000  FORMAT (/ QUARTER END ROUTINE FOR QUARTER ', I1)
```

FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE YEARAND(Y)

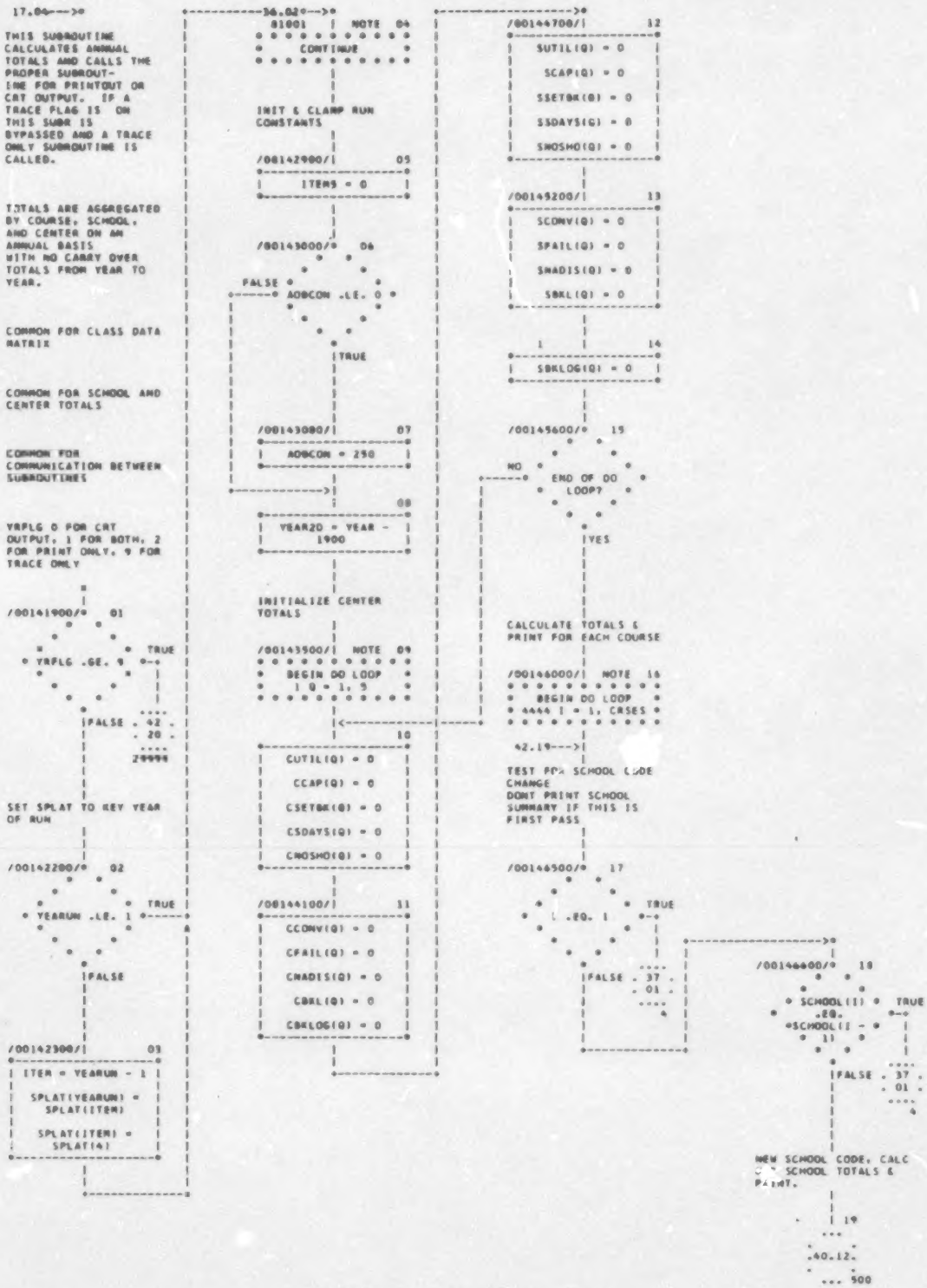


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE YEARND(Y)

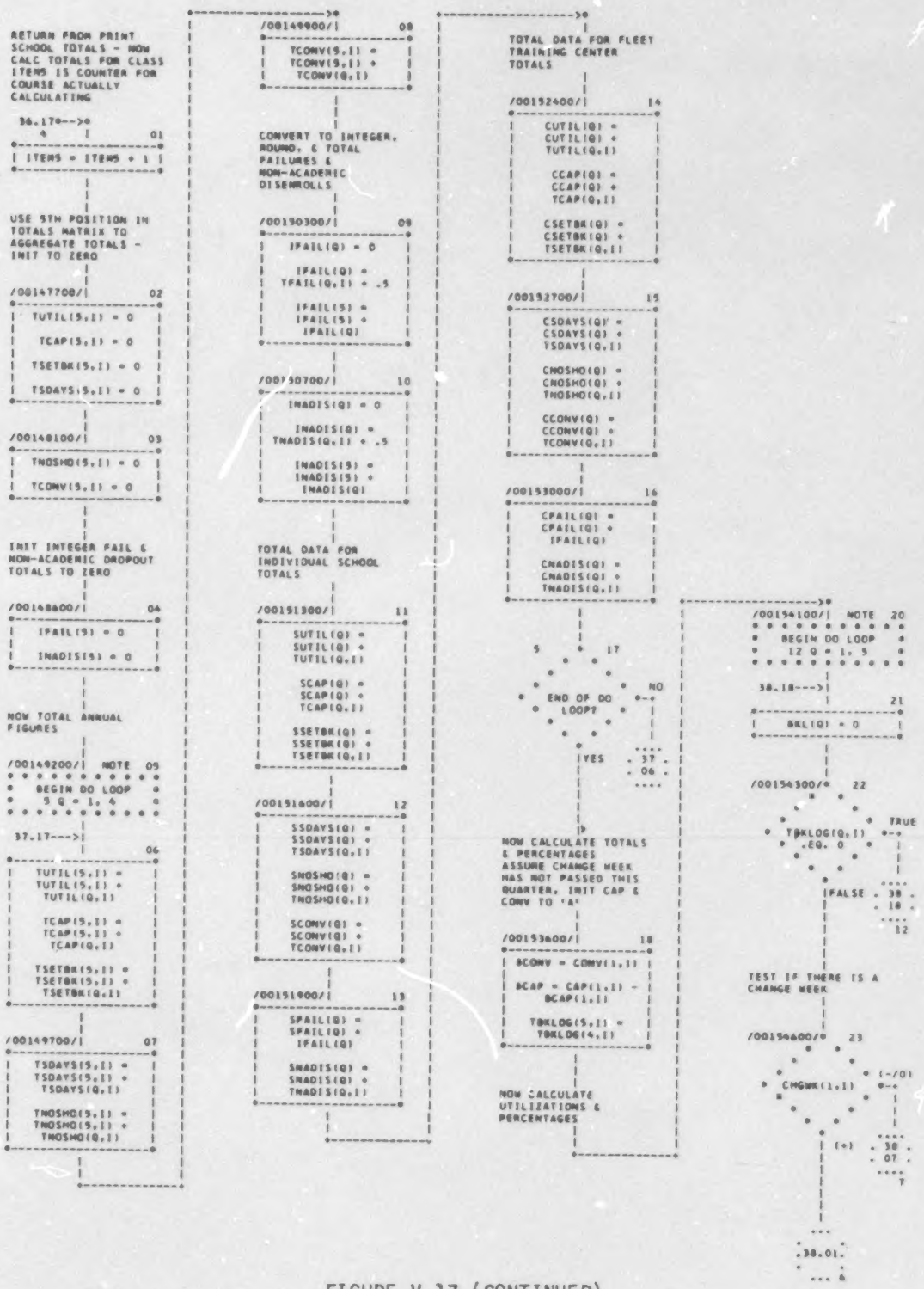


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE YEARND(Y)

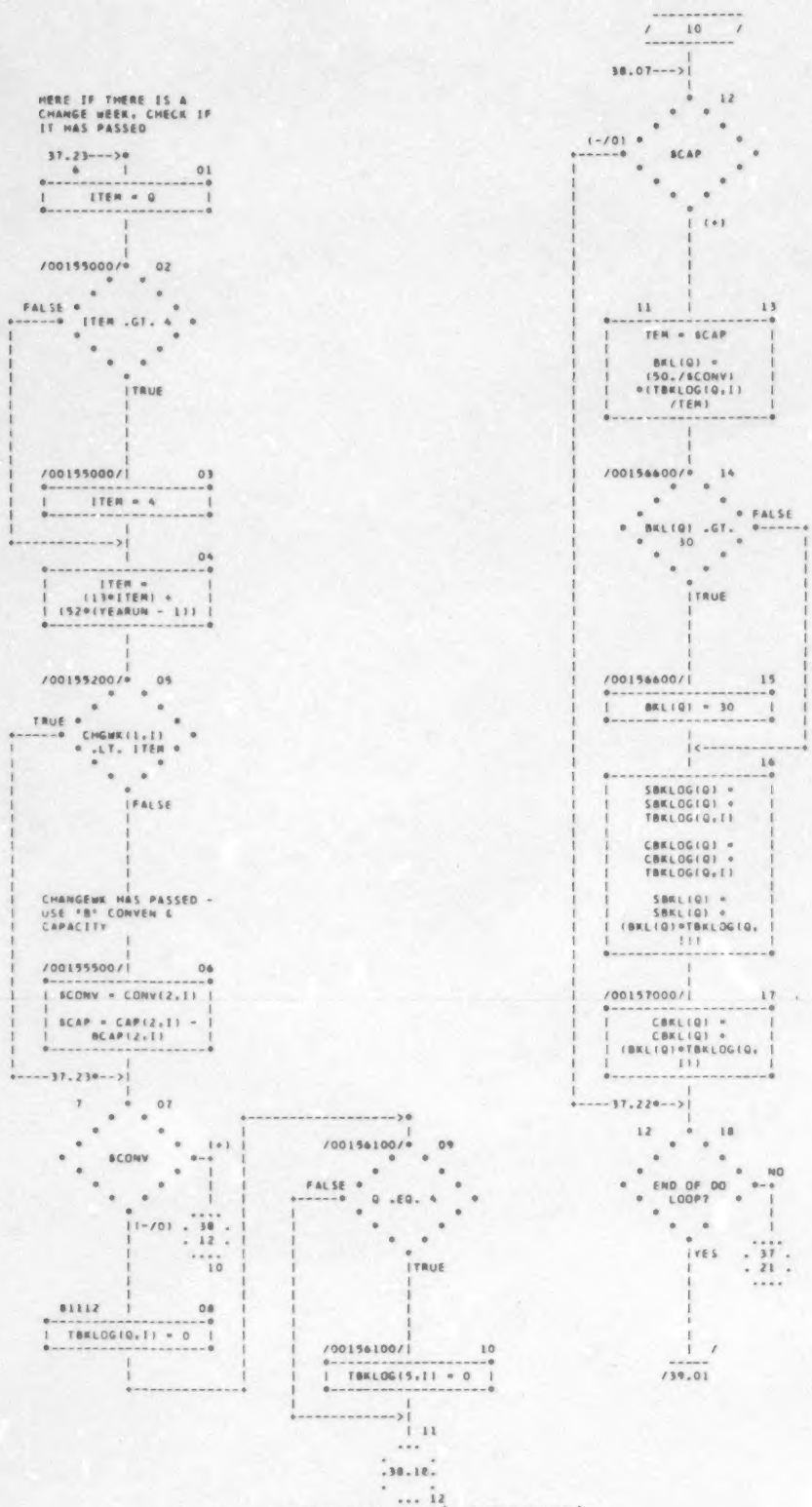


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE YEARND(Y)

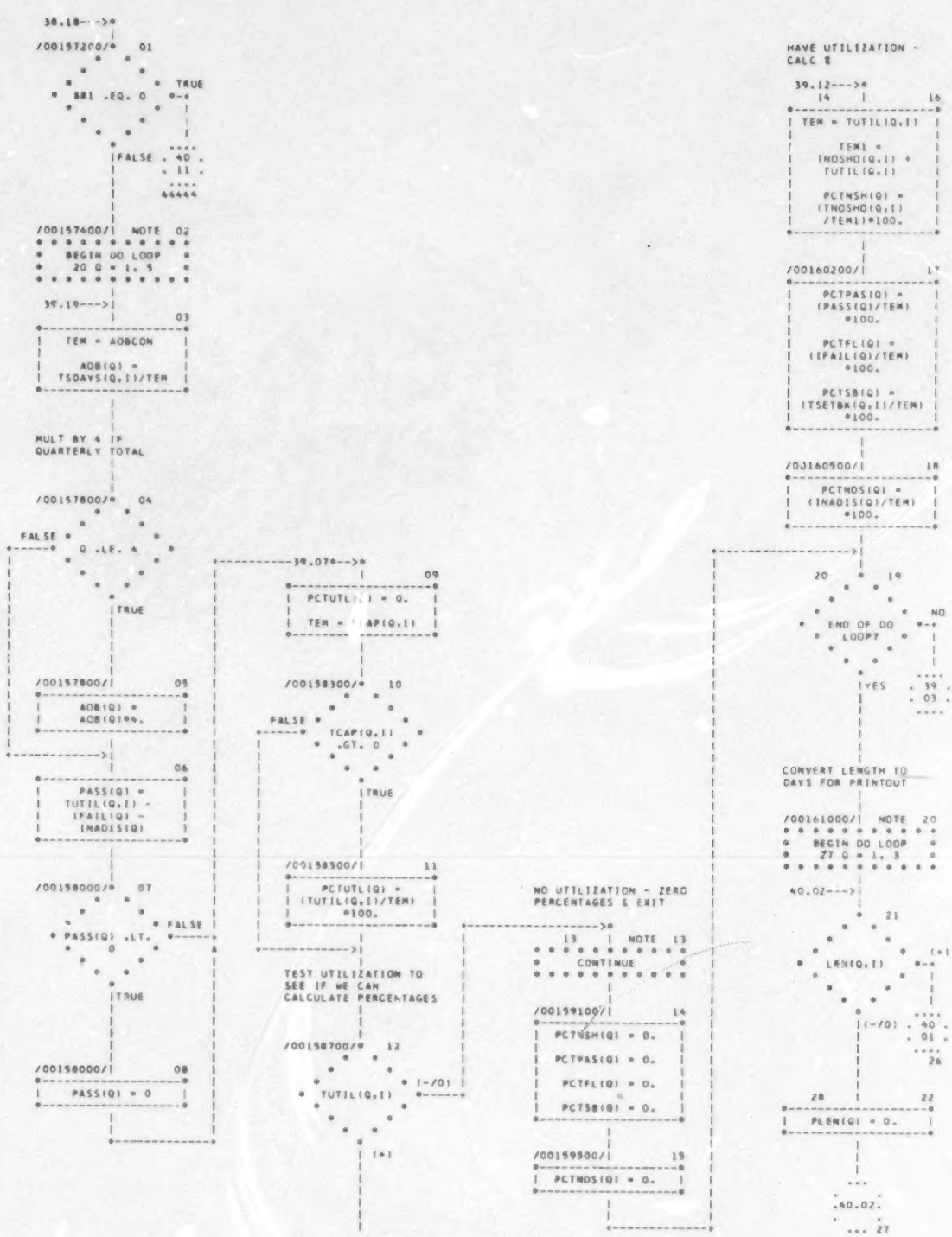


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE YEARND(Y)

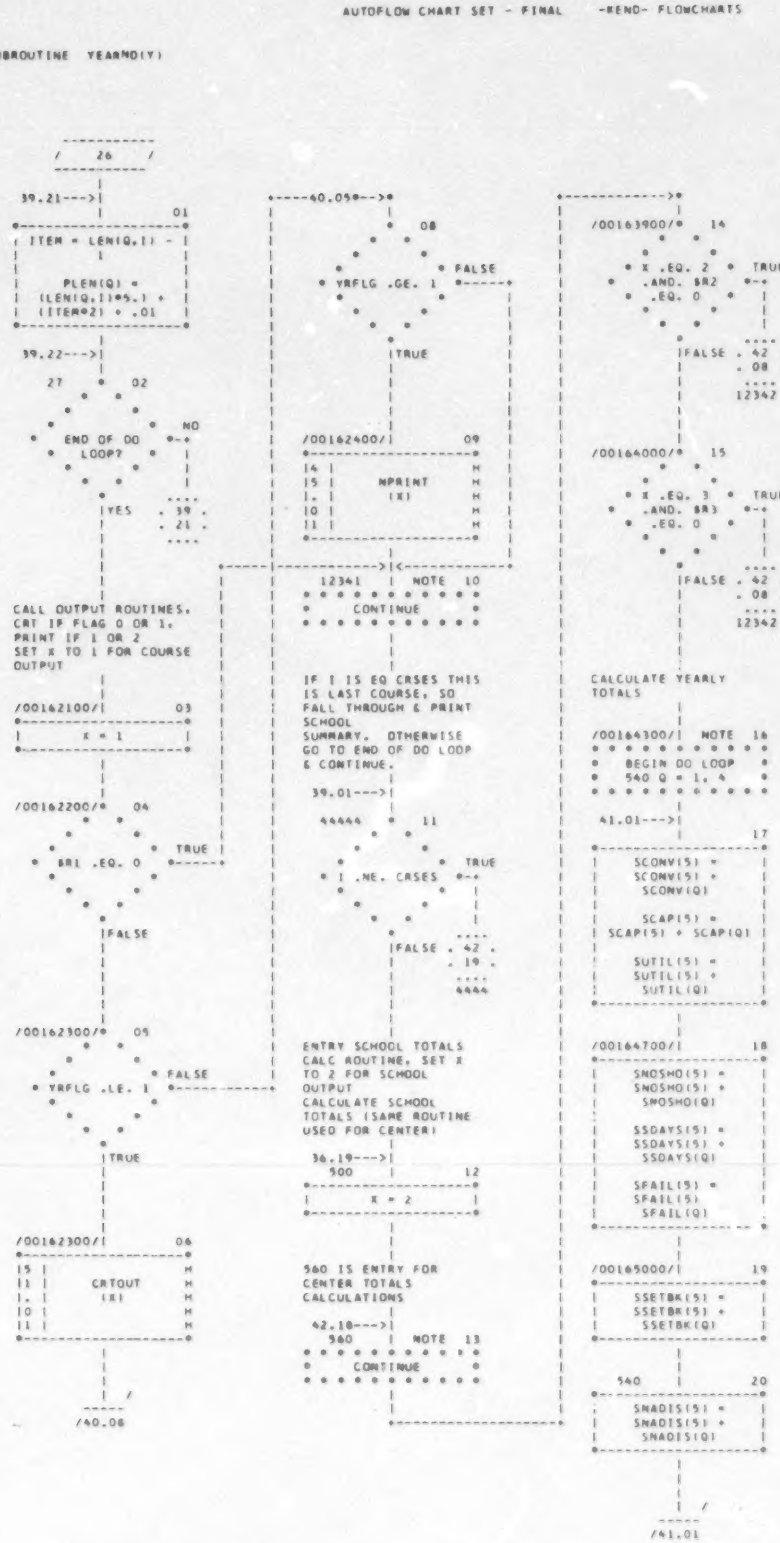


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE YEARND(Y)

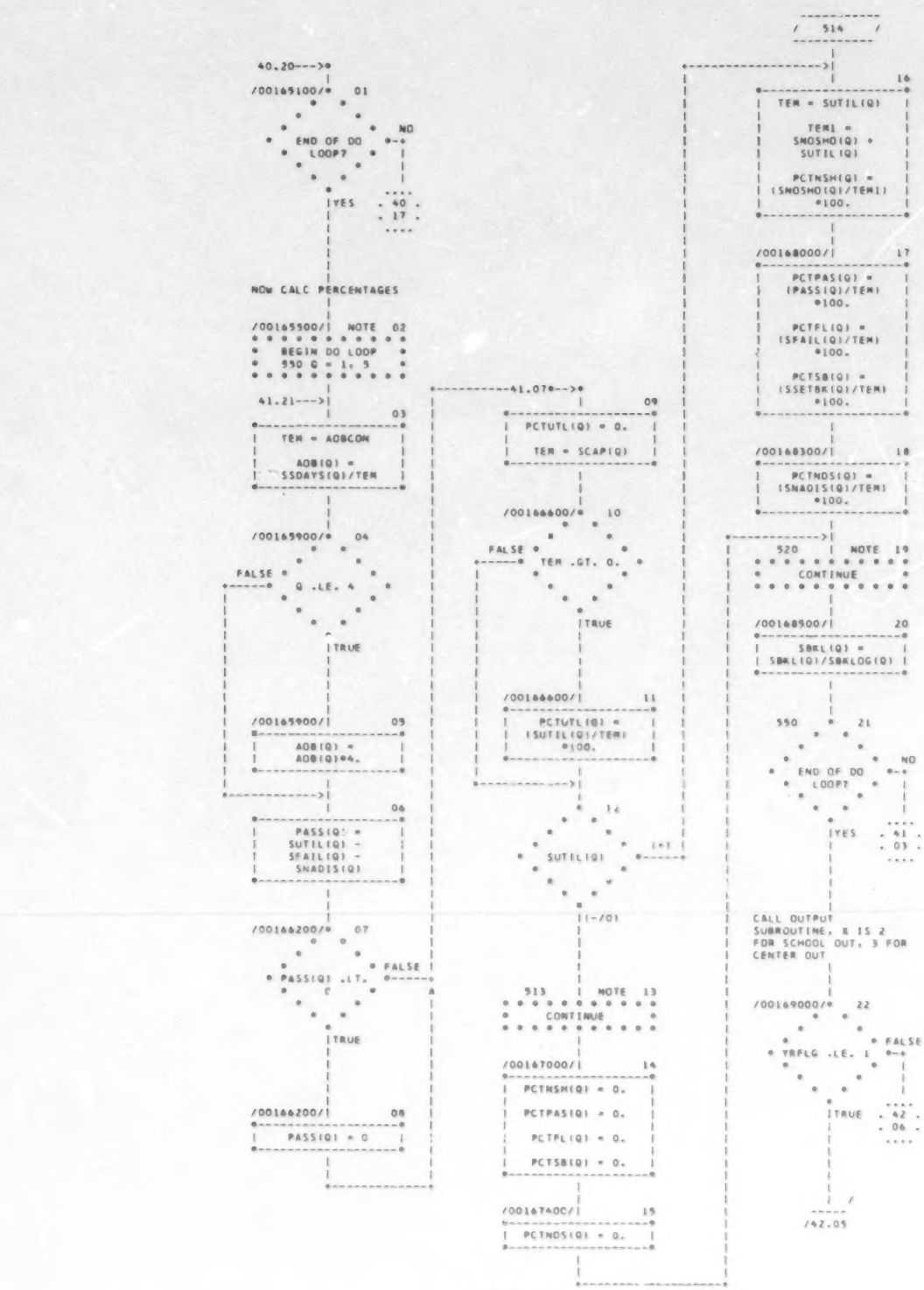


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE YEARNO(Y)

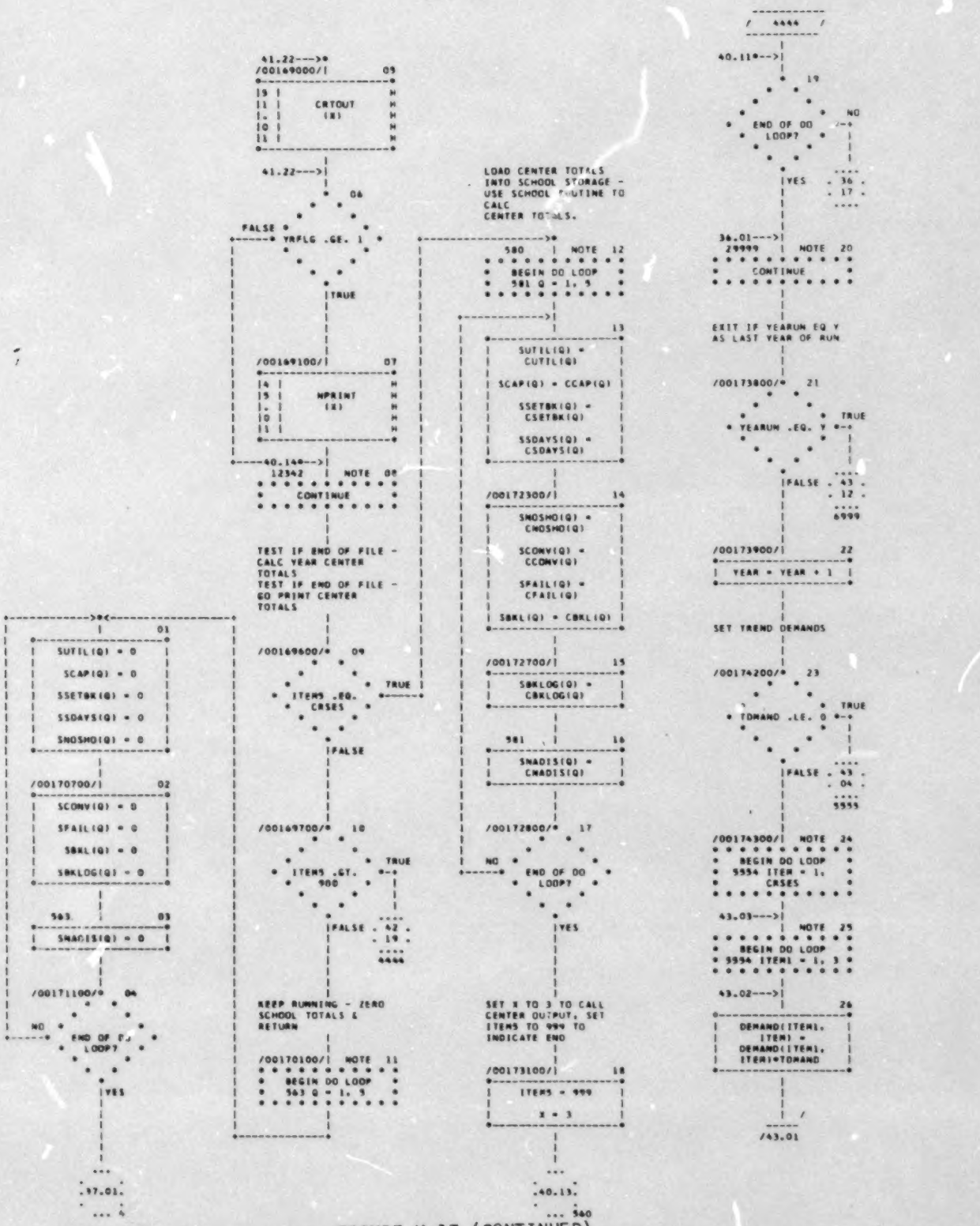


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE YEARND(Y)

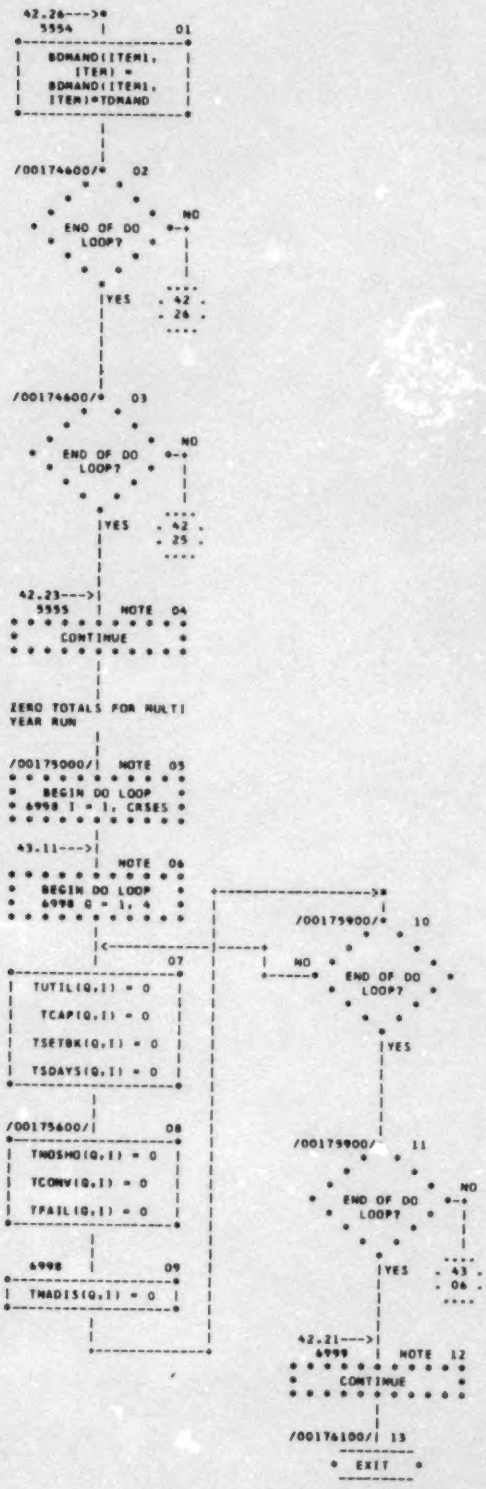


FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
IMPLICIT INTEGER*2 (A-Z,$)
REAL*4 LEN, NOSHOW, NOSHPL, BXLGPL, BPOOL, NADIS, FAIL, CDP
REAL*4 TDMAND, SETBAK
COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
COMMON /DATA/ BAXLOG (176), NOSHOW (176), BXLGPL (176)
COMMON /DATA/ BXLG (176), TDMAND, SETBAK
INTEGER*4 JNO, PNEC, SCHOOL
REAL*4 PAOB, PFAIL
REAL*8 NAME1, NAME2, CATNO1
COMMON /TEXT/ JNO (176), NAME1 (176), NAME2 (176), CATNO1 (176)
COMMON /TEXT/  CATNO2 (176), PNEC (176), TYPE (176), PAOB (176)
COMMON /TEXT/ PFAIL (176), PRTY (176), SCHOOL (176)
REAL*4 TFAIL, TNADIS
INTEGER*4 TSDAYS, TCAP, TUTIL
COMMON /TOTALS/ TCAP (5, 176),  TUTIL (5, 176),  TSETBK (5, 176)
COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
COMMON /TOTALS/ TNADIS (5, 176), TBXLOG (5, 176), TCONV (5, 176)
INTEGER*4 SUTIL, CUTIL, SCAP, CCAP, SSDAYS, CSDAYS, SFAIL, SNADIS
INTEGER*4 PASS, SBXLOG, CBXLOG, SBXL,CBXL
REAL*4 AOB, PCTUTL, PCTNSH, PCTPAS, PCTPL, PCTSB, PCTNDS, PLEN
COMMON /T/ SUTIL (5), SCAP (5), SSDAYS (5), SSETBK (5), SNOSHO (5)
COMMON /T/ CUTIL (5), CCAP (5), CSDAYS (5), CSETBK (5), CNOSHO (5)
COMMON /T/SBXL(5),SBXLOG(5),SCONV (5), SFAIL (5), SNADIS (5)
COMMON /T/ CCONV(5),CBXL(5),CBXLOG(5), CFAIL (5), CNADIS (5)
COMMON /T/ PCTUTL (5), PCTNSH (5), PCTNDS (5), INADIS (5)
COMMON /T/ AOB (5), PASS (5), PCTPAS (5), PCTPL (5), PCTSB (5)
COMMON /T/ IFAIL (5), PLEN (3), BXL (5), SPLAT (4), YEAR2D
INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, BLEN
COMMON DUMMY, BCONV, BLEN, BCAP, BBCAP, LAST, COURSE
COMMON BR1,BR2,BR3,BR4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON
COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
```

FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE NPRINT(X)

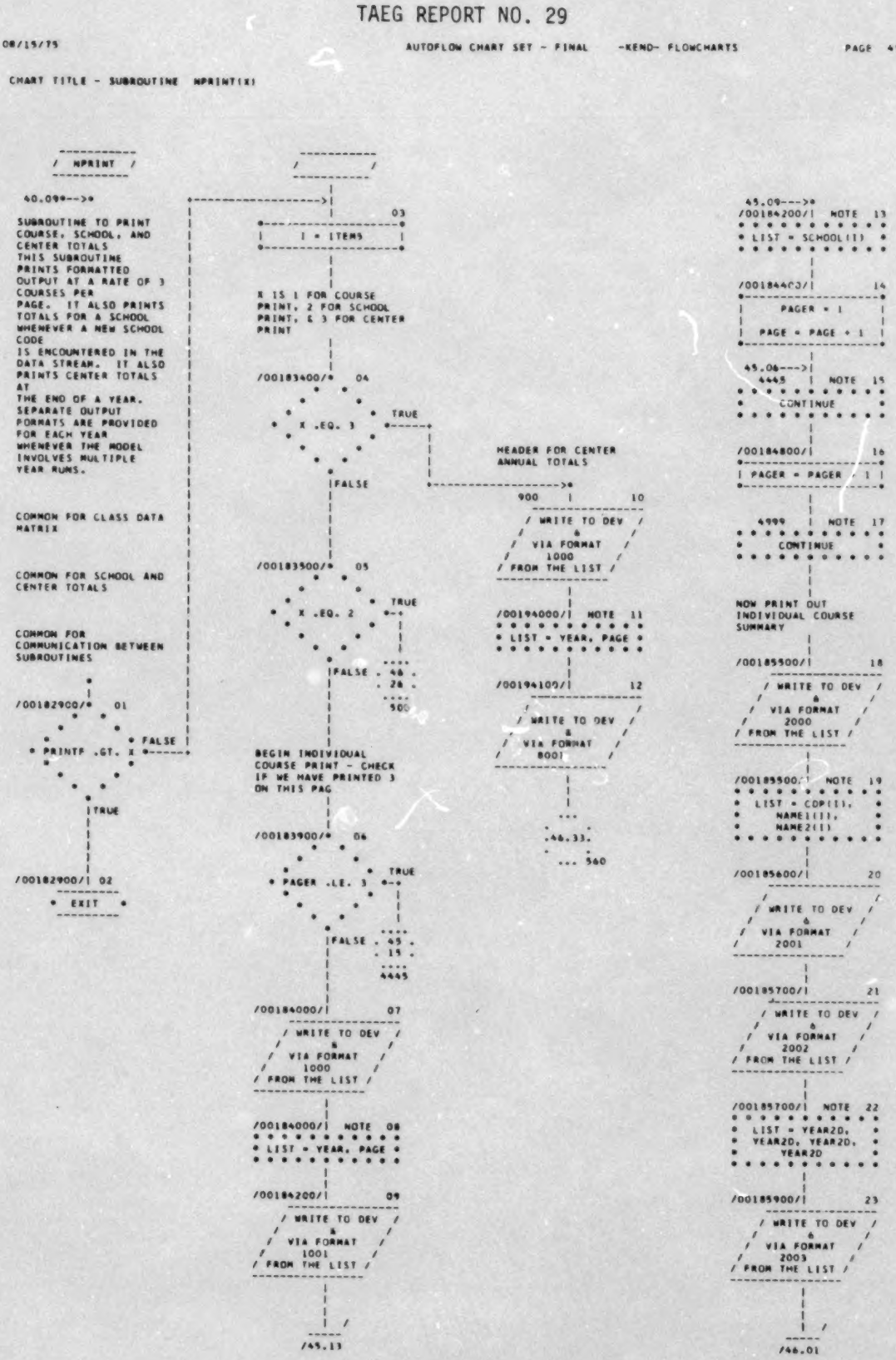


FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE NPRINT(X)

```
 45.23--->*
/00185900/|  NOTE  01
* * * * * * * * * * *
*  LIST = JNO(I),   *
*    CATNO1(I),     *
*    CATNO2(I),     *
* PNEC(I), TYPE(I), *
*      PRTY(I)      *
* * * * * * * * * * *
        |
        |
/00186200/|        02
   ----------------
    / WRITE TO DEV  /
   /       &        /
  /  VIA FORMAT    /
 /     2004       /
/ FROM THE LIST /
----------------
        |
        |
/00186200/|  NOTE  03
* * * * * * * * * * *
*      LIST =       *
*    TCONV(1,I),    *
*    TCONV(2,I),    *
*    TCONV(3,I),    *
*    TCONV(4,I),    *
*    TCONV(5,I)     *
* * * * * * * * * * *
        |
        |
/00186500/|        04
   ----------------
    / WRITE TO DEV  /
   /       &        /
  /  VIA FORMAT    /
 /     2005       /
/ FROM THE LIST /
----------------
        |
        |
/00186500/|  NOTE  05
* * * * * * * * * * *
* LIST = TCAP(1,I), *
*    TCAP(2,I),     *
*    TCAP(3,I),     *
*    TCAP(4,I),     *
*    TCAP(5,I)      *
* * * * * * * * * * *
        |
        |
/00186800/|        06
   ----------------
    / WRITE TO DEV  /
   /       &        /
  /  VIA FORMAT    /
 /     2006       /
/ FROM THE LIST /
----------------
        |
        |
/00186800/|  NOTE  07
* * * * * * * * * * *
*  LIST = PAOB(I),  *
*      PFAIL(I),    *
*      BKLG(I),     *
*    BKLOG(I),      *
*  (TUTIL(Q,I),     *
*  PCTUTL(Q),Q =    *
*       1,5)        *
* * * * * * * * * * *
        |
        |
/00187100/|        08
   ----------------
    / WRITE TO DEV  /
   /       &        /
  /  VIA FORMAT    /
 /     2007       /
/ FROM THE LIST /
----------------
        |
        |
/00187100/|  NOTE  09
* * * * * * * * * * *
* LIST = SPLAT(I),  *
*      SPLAT(2),    *
*      SPLAT(3),    *
*  (TNOSHO(Q,I),    *
*  PCTNSH(Q),Q =    *
*       1,5)        *
* * * * * * * * * * *
        |
        |
```

```
*------------------>*
|        /00187400/|        10
|          ----------------
|           / WRITE TO DEV  /
|          /       &        /
|         /  VIA FORMAT    /
|        /     2008       /
|       / FROM THE LIST /
|       ----------------
|               |
|               |
|       /00187400/|  NOTE  11
|       * * * * * * * * * * *
|       *      LIST =       *
|       *   (TBKLOG(Q,I),   *
|       *  BKL(Q),Q = 1,5)  *
|       * * * * * * * * * * *
|               |
|               |
|       /00187600/|        12
|          ----------------
|           / WRITE TO DEV  /
|          /       &        /
|         /  VIA FORMAT    /
|        /     2009       /
|       / FROM THE LIST /
|       ----------------
|               |
|               |
|       /00187600/|  NOTE  13
|       * * * * * * * * * * *
|       * LIST = CONV(1,I), *
|       *    CONV(2,I),     *
|       *    CONV(3,I),     *
|       *   (TSDAYS(Q,I),   *
|       *  ADB(Q),Q = 1,5)  *
|       * * * * * * * * * * *
|               |
|               |
|       /00187900/|        14
|          ----------------
|           / WRITE TO DEV  /
|          /       &        /
|         /  VIA FORMAT    /
|        /     2010       /
|       / FROM THE LIST /
|       ----------------
|               |
|               |
|       /00187900/|  NOTE  15
|       * * * * * * * * * * *
|       *  LIST = PLEN(I),  *
|       * PLEN(2), PLEN(3)  *
|       * * * * * * * * * * *
|               |
|               |
|       /00188100/|        16
|          ----------------
|           / WRITE TO DEV  /
|          /       &        /
|         /  VIA FORMAT    /
|        /     2011       /
|       / FROM THE LIST /
|       ----------------
|               |
|               |
|       /00188100/|  NOTE  17
|       * * * * * * * * * * *
|       * LIST = CAP(1,I),  *
|       *     CAP(2,I),     *
|       *     CAP(3,I),     *
|       *     (PASS(Q),     *
|       *   PCTPAS(Q),Q =   *
|       *       1,5)        *
|       * * * * * * * * * * *
|               |
|               |
|       /00188400/|        18
|          ----------------
|           / WRITE TO DEV  /
|          /       &        /
|         /  VIA FORMAT    /
|        /     2012       /
|       / FROM THE LIST /
|       ----------------
|               |
|               |
|       /00188400/|  NOTE  19
|       * * * * * * * * * * *
|       * LIST = BCAP(1,I), *
|       *     BCAP(2,I),    *
|       *     BCAP(3,I),    *
|       *     (IFAIL(Q),    *
|       * PCTFL(Q),Q = 1,5) *
|       * * * * * * * * * * *
|               |
|               |
*---------------*
```

```
*------------------>*
|        /00188700/|        20
|          ----------------
|           / WRITE TO DEV  /
|          /       &        /
|         /  VIA FORMAT    /
|        /     2013       /
|       / FROM THE LIST /
|       ----------------
|               |
|               |
|       /00188700/|  NOTE  21
|       * * * * * * * * * * *
|       *      LIST =       *
|       *   DEMAND(1,I),    *
|       *   DEMAND(2,I),    *
|       *   DEMAND(3,I),    *
|       *   (TSETBK(Q,I),   *
|       * PCTSB(Q),Q = 1,5) *
|       * * * * * * * * * * *
|               |
|               |
|       /00189000/|        22
|          ----------------
|           / WRITE TO DEV  /
|          /       &        /
|         /  VIA FORMAT    /
|        /     2014       /
|       / FROM THE LIST /
|       ----------------
|               |
|               |
|       /00189000/|  NOTE  23
|       * * * * * * * * * * *
|       *      LIST =       *
|       *   BDMAND(1,I),    *
|       *   BDMAND(2,I),    *
|       *   BDMAND(3,I),    *
|       *    (INADIS(Q),    *
|       *   PCTNDS(Q),Q =   *
|       *       1,5)        *
|       * * * * * * * * * * *
|               |
|               |
|       /00189300/|        24
|          ----------------
|           / WRITE TO DEV  /
|          /       &        /
|         /  VIA FORMAT    /
|        /     2015       /
|       / FROM THE LIST /
|       ----------------
|               |
|               |
|       /00189300/|  NOTE  25
|       * * * * * * * * * * *
|       * LIST = OFFSET(I), *
|       *     CHGWK(1,I),   *
|       *     CHGWK(2,I)    *
|       * * * * * * * * * * *
|               |
|               |
|               |
|               |
|              ...
|             .   .
|            .47.24.
|             .   .
|              ... 4444
```

SCHOOL TOTALS ROUTINE

```
 45.05--->*
   500    |        26
   ----------------
    / WRITE TO DEV  /
   /       &        /
  /  VIA FORMAT    /
 /     1000       /
/ FROM THE LIST /
----------------
        |
        |
/00190000/|  NOTE  27
* * * * * * * * * * *
* LIST = YEAR, PAGE *
* * * * * * * * * * *
        |
        |
/00190100/|        28
*-----------------*
|       J = 1     |
|                 |
| PAGE = PAGE + 1 |
|                 |
|    PAGER = 4    |
*-----------------*
        |
        |
/00190500/|        29
   ----------------
    / WRITE TO DEV  /
   /       &        /
  /  VIA FORMAT    /
 /     1001       /
/ FROM THE LIST /
----------------
        |
        |
/00190500/|  NOTE  30
* * * * * * * * * * *
* LIST = SCHOOL(J)  *
* * * * * * * * * * *
        |
        |
/00190600/|        31
   ----------------
    / WRITE TO DEV  /
   /       &        /
  /  VIA FORMAT    /
 /     7001       /
/ FROM THE LIST /
----------------
        |
        |
/00190600/|  NOTE  32
* * * * * * * * * * *
* LIST = SCHOOL(J)  *
* * * * * * * * * * *
        |
        |
ENTRY FOR FTC TOTALS
FROM HERE ON SAME
ROUTINE IS USED FOR
SCHOOL AND CENTER
TOTALS PRINT
ROUTINE.  YEAR DATA
HAS BEEN OVERWRITTEN
INTO SCHOOL STORAGE
AREAS.
        |
 45.12--->|
   560    |  NOTE  33
* * * * * * * * * * *
*     CONTINUE      *
* * * * * * * * * * *
        |
        |
PRINT ROUTINE
PRINT SCHOOL AND
CENTER TOTALS
        |
/00192000/|        34
   ----------------
    / WRITE TO DEV  /
   /       &        /
  /  VIA FORMAT    /
 /     7002       /
/ FROM THE LIST /
----------------
        |
        |
        |
        |  /
       -----
      /47.01
```

FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE NPRINT(X)

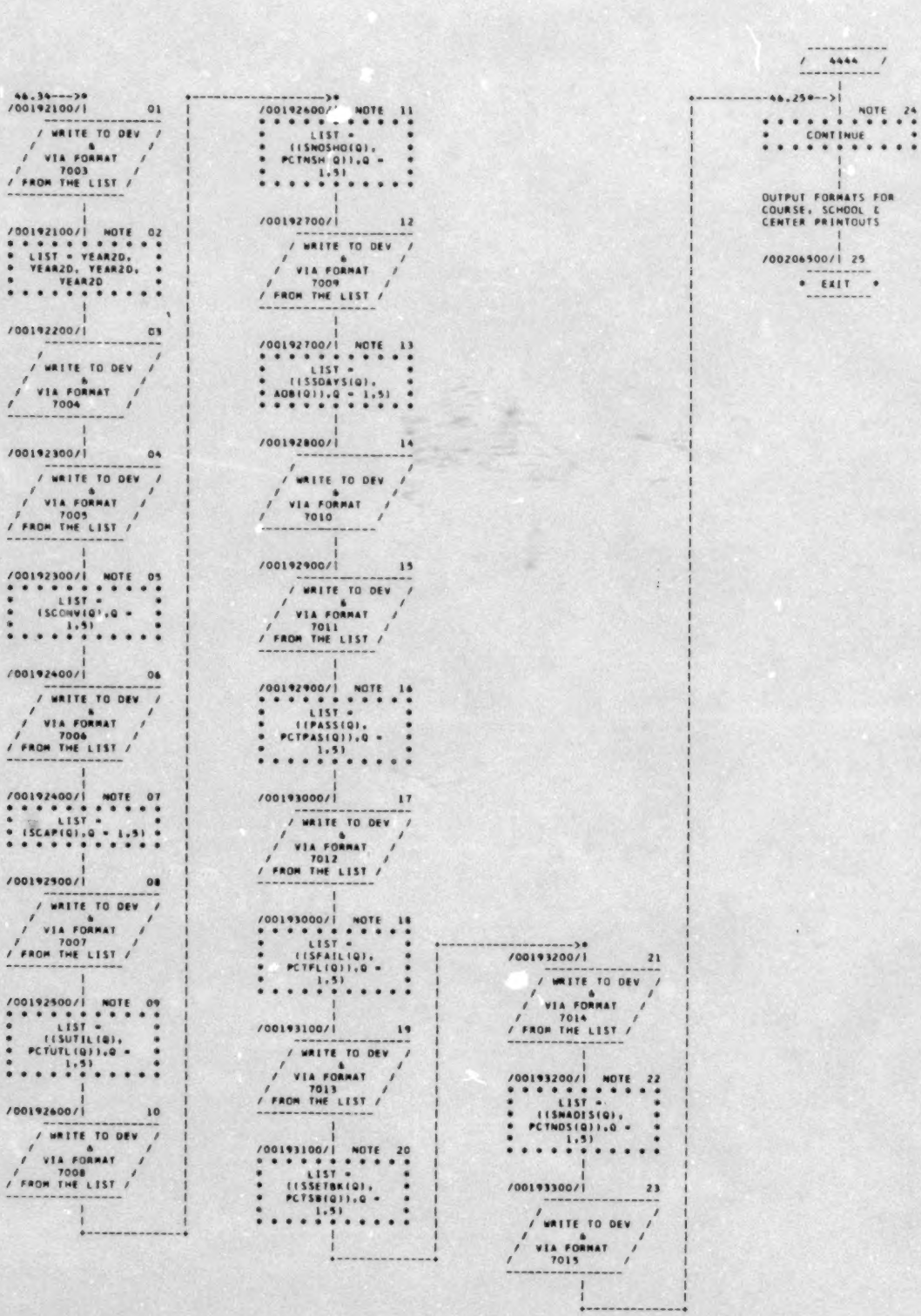


FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATE. ,TS

```
      IMPLICIT INTEGER*2 (A-Z,$)
      REAL*4 LEN, NOSHOW, NOSHPL, BXLGPL, BPOOL, NADIS, FAIL, CDP
      REAL*4 TDMAND, SETBAK
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BXLGPL (176)
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK
      INTEGER*4 JNO, PNEC, SCHOOL
      REAL*4 PAOB, PFAIL
      REAL*8 NAME1, NAME2, CATNO1
      COMMON /TEXT/ JNO (176), NAME1 (176), NAME2 (176), CATNO1 (176)
      COMMON /TEXT/  CATNO2 (176), PNEC (176), TYPE (176), PAOB (176)
      COMMON /TEXT/ PFAIL (176), PRTY (176), SCHOOL (176)
      REAL*4 TFAIL, TNADIS
      INTEGER*4 TSDAYS, TCAP, TUTIL
      COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176)
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
      COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
      INTEGER*4 SUTIL, CUTIL, SCAP, CCAP, SSDAYS, CSDAYS, IFAIL, INADIS
      INTEGER*4 PASS, SBKLOG, CBKLOG, SBKL,CBKL
      REAL*4 AOB, PCTUTL, PCTNSH, PCTPAS, PCTFL, PCTSB, PCTNDS, PLEN
      COMMON /T/ SUTIL (5), SCAP (5), SSDAYS (5), SSETBK (5), SNOSHO (5)
      COMMON /T/ CUTIL (5), CCAP (5), CSDAYS (5), CSETBK (5), CNOSHO (5)
      COMMON /T/SBKL(5),SBKLOG(5),SCONV (5), SFAIL (5), SNADIS (5)
      COMMON /T/ CCONV(5),CBKL(5),CBKLOG(5), CFAIL (5), CNADIS (5)
      COMMON /T/ PCTUTL (5), PCTNSH (5), PCTNDS (5), INADIS (5)
      COMMON /T/ AOB (5), PASS (5), PCTPAS (5), PCTFL (5), PCTSB (5)
      COMMON /T/ IFAIL (5), PLEN (3), BKL (5), SPLAT (4), YEAR2D
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, BLEN
      COMMON DUMMY, BCONV, BLEN, BCAP, BBCAP, LAST, COURSE
      COMMON BR1,BR2,BR3,BR4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
      COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
      COMMON YEAR, CASES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON
      COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
      DATA PAGE /1/, PAGER /4/
8001  FORMAT (1H0, T42, '******** TOTALS FOR FLEET TRAINING CENTER *****
      ***', //////)
1000  FORMAT (1H1,T49,'TPF MODEL -- FISCAL YEAR ',I4,' SUMMARY',T123,
       'PAGE ',I3)
1001  FORMAT (' ', T58, 'SCHOOL CODE  ', A4)
2000  FORMAT (// ' ', A4, 5X, 2A8)
2001  FORMAT (' ', T78, 'QUARTERLY PROJECTIONS')
2002  FORMAT (' JNO   CATALOG NO   NEC   TYPE  PRTY', T60,'| 1 QTR FY ',
       I2,' | 2 QTR FY ',I2,' | 3 QTR FY ',I2,' | 4 QTR FY ',I2,
       ' | ANNUAL TOTAL |')
```

FIGURE V-17 (CONTINUED)

```
2003    FORMAT (1X, A4, 2X, A8, A2, 2X, A4, 4X, A2, 4X, A2,
                                    T42,   '-------------------|-----------
        --|-------------|-------------|-------------|--------------|')
2004    FORMAT (1X, T42, 'NUMBER CONVENINGS |', 4 (I5, ' |', 6X, '|'),
         I6, ' |', 6X, '|')
2005    FORMAT (' P AOB  PFAIL   BACKLOG  STU  LENGTH',
                    T42, 'COURSE CAPACITY   |', 4 (I5, ' |', 6X, '|'),
         I6, ' |', 6X, '|')
2006    FORMAT (1X, F5.1, F6.1, '%    INIT', 2X, I5, I4, ' WKS',
                    T42, 'UTILIZATION       |', 4 (I5, ' |', F5.1, '%|'),
         I6, ' |', F5.1, '%|')
2007    FORMAT (1X, T19, 3 (4X, A2),
                    T42, 'NO SHOWS          |', 4 (I5, ' |', F5.1, '%|'),
         I6, ' |', F5.1, '%|')
2008    FORMAT (' --SCHEDULE DATA--    FY  FY+1  FY+2',
                    T42, 'BKLOG-STUDENTS/WKS|', 4 (I5, ' |', I2, ' WKS|'),
         I6, ' |', I2, ' WKS|')
2009    FORMAT (' ANNUAL CONVENINGS', 3I6,
                    T42, 'STUDENT DAYS / AOB|', 4 (I5, ' |', F5.1, ' |'),
         I6, ' |', F5.1, ' |')
2010    FORMAT (' COURSE LENGTH DYS', 3F6.1,
         T60,'|', T67,'|', T74,'|', T81,'|', T88,'|', T95,'|', T102,'|',
         T109,'|', T116,'|', T124,'|', T131,'|')
2011    FORMAT (' COURSE CAPACITY  ', 3I6,
                    T42, 'TOTAL PASS        |', 4 (I5, ' |', F5.1, '%|'),
         I6, ' |', F5.1, '%|')
2012    FORMAT (' BUPERS SEATS     ', 3I6,
                    T42, 'TOTAL FAIL        |', 4 (I5, ' |', F5.1, '%|'),
         I6, ' |', F5.1, '%|')
2013    FORMAT (' ANNUAL DEMAND    ', 3I6,
                    T42, 'ACADEMIC SETBACK  |', 4 (I5, ' |', F5.1, '%|'),
         I6, ' |', F5.1, '%|')
2014    FORMAT (' BUPERS DEMAND    ', 3I6,
                    T42, 'NON-ACAD DISENROLL|', 4 (I5, ' |', F5.1, '%|'),
         I6, ' |', F5.1, '%|')
2015    FORMAT (' OFFSET / CHG WEEK', 3I6,
                  T42,  '-------------------------------------------------
        ------------------------------------------' ,//)
7001    FORMAT (//////1X,T17,'SCHOOL CODE ',A4,' SUMMARY')
7002    FORMAT (//1X,T59,'QUARTERLY PROJECTIONS')
7003    FORMAT (1X,T37,'|  1 QTR FY ',I2,'  |  2 QTR FY ',I2,'  |  3 QTR
        FY ',I2,' |  4 QTR FY ',I2,'  |  ANNUAL TOTAL  |')
7004    FORMAT (1X,T17,'--------------------|---------------|-------------
        --|---------------|---------------|----------------|')
7005    FORMAT (1X,T17,'NUMBER OF CONVENINGS|',I6,' |',7X,'|',I6,' |',7X,
        '|',I6,' |',7X,'|',I6,' |',7X,'|',I7,' |',7X,'|')
7006    FORMAT (1X,T17,'COURSE CAPACITY',5X,'|',I6,' |',7X,'|',I6,' |',
         7X,'|',I6,' |',7X,'|',I6,' |',7X,'|',I7,' |',7X,'|')
```

FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
7007    FORMAT (1X,T17,'UTILIZATION',T37,'|',I6,' |',F6.1,'%|',I6,' |',
         F6.1,'%|',I6,' |',F6.1,'%|',I6,' |',F6.1,'%|',I7,' |',F6.1,
         '%|')
7008    FORMAT (1X,T17,'NO SHOWS',T37,'|',I6,' | ',F5.1,'%|',I6,' | ',
         F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',F5.1,
         '%|')
7009    FORMAT (1X,T17,'STUDENT DAYS / ADB  |',I6,' |',F6.1,' |',I6,' |',
         F6.1,' |',I6,' |',F6.1,' |',I6,' |',F6.1,' |',I7,' |',
         F6.1,' |')
7010    FORMAT (1X,T37,'|',7X,'|',7X,'|',7X,'|',7X,'|',7X,'|',7X,'|',7X,
         '|',7X,'|',8X,'|',7X,'|')
7011    FORMAT (1X,T17,'TOTAL PASS',T37,'|',I6,' | ',F5.1,'%|',I6,' | ',
         F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',F5.1,
         '%|')
7012    FORMAT (1X,T17,'TOTAL FAIL',T37,'|',I6,' | ',F5.1,'%|',I6,' | ',
         F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',F5.1,
         '%|')
7013    FORMAT(1X,T17,'ACADEMIC SETBACK    |',I6,' | ',F5.1,'%|',I6,
         ' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',
         F5.1,'%|')
7014    FORMAT (1X,T17,'NON-ACADEM DISENROLL|',I6,' | ',F5.1,'%|',I6,
         ' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',
         F5.1,'%|')
7015    FORMAT (1X,T17,'-----------------------------------------------------
        ----------------------------------------------------')
```

FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE CRTOUT(X)

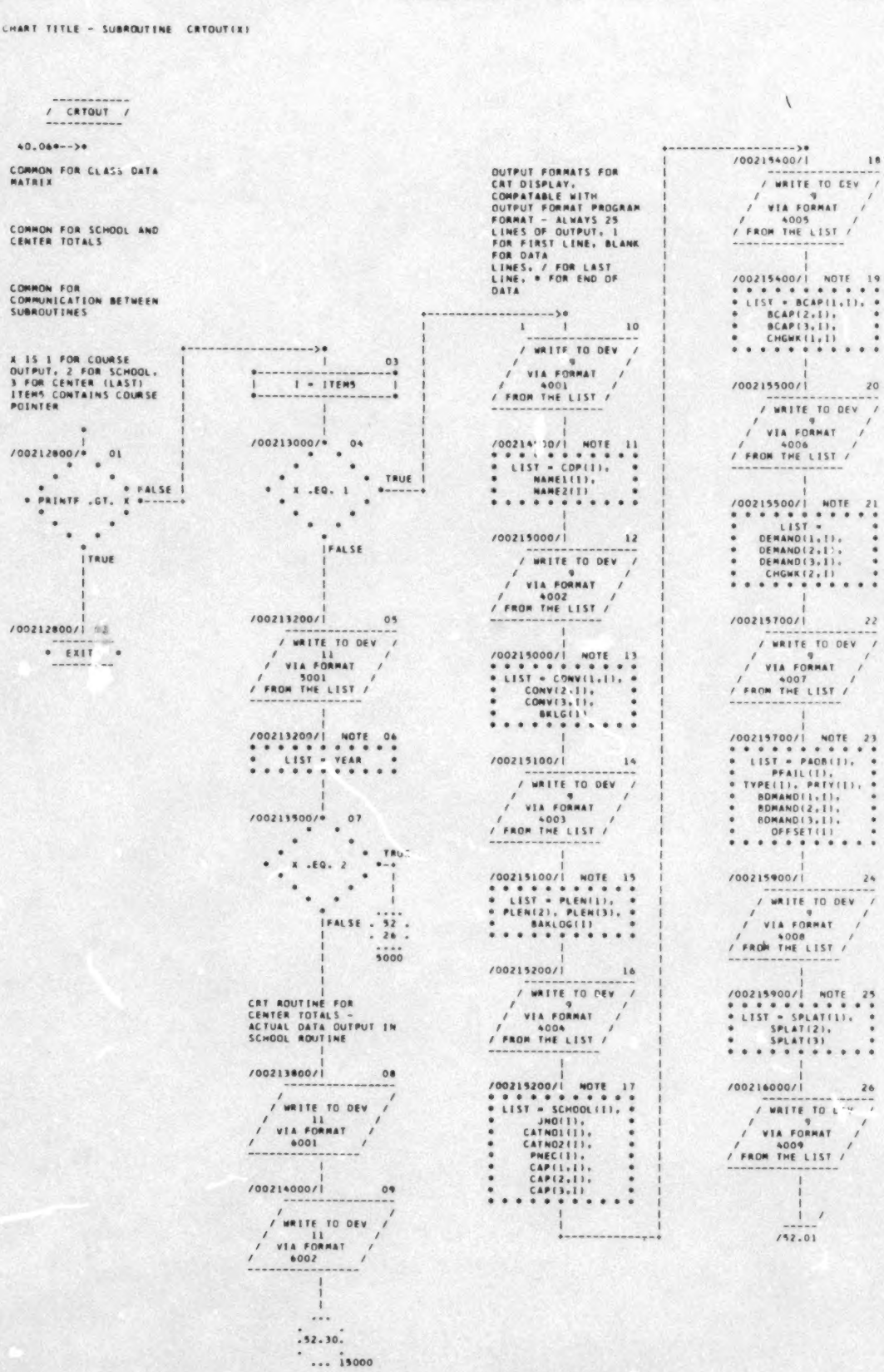


FIGURE V-17 (CONTINUED)

```
CHAR/ TITLE - SUBROUTINE  CRTOU(IX)

 51.26--->*
/00216000/|  NOTE  01
* * * * * * * * * * *
*  LIST = YEAR2D,    *
*  YEAR2D, YEAR2D,   *
*      YEAR2D        *
* * * * * * * * * * *
          |
          |
/00216100/|        02
    ----------------
   /                /
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4010      /
----------------
          |
          |
/00216200/|        03
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4011      /
/ FROM THE LIST /
----------------
          |
          |
/00216200/|  NOTE  04
* * * * * * * * * * *
*      LIST =       *
*    TCONV(1,I),    *
*    TCONV(2,I),    *
*    TCONV(3,I),    *
*    TCONV(4,I),    *
*    TCONV(5,I)     *
* * * * * * * * * * *
          |
          |
/00216400/|        05
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4012      /
/ FROM THE LIST /
----------------
          |
          |
/00216400/|  NOTE  06
* * * * * * * * * * *
* LIST = TCAP(1,I), *
*    TCAP(2,I),     *
*    TCAP(3,I),     *
*    TCAP(4,I),     *
*    TCAP(5,I)      *
* * * * * * * * * * *
          |
          |
/00216600/|        07
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4013      /
/ FROM THE LIST /
----------------
          |
          |
/00216600/|  NOTE  08
* * * * * * * * * * *
*      LIST =       *
*   (ITUTIL(Q,I),   *
*  PCTUTL(Q)),Q =   *
*       1,5)        *
* * * * * * * * * * *
          |
          |
/00216700/|        09
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4014      /
/ FROM THE LIST /
----------------
          |
          |
          |
          +--------------*

          +-----------------*
          |
          |
/00216700/|  NOTE  10
* * * * * * * * * * *
*      LIST =       *
*   (ITNOSHO(Q,I),  *
*   PCTNSH(Q)),Q =  *
*       1,5)        *
* * * * * * * * * * *
          |
          |
/00216800/|        11
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4015      /
/ FROM THE LIST /
----------------
          |
          |
/00216800/|  NOTE  12
* * * * * * * * * * *
*      LIST =       *
*   (ITBKLOG(Q,I),  *
*  BKL(Q)),Q = 1,5) *
* * * * * * * * * * *
          |
          |
/00216900/|        13
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4016      /
/ FROM THE LIST /
----------------
          |
          |
/00216900/|  NOTE  14
* * * * * * * * * * *
*      LIST =       *
*   (ITSDAYS(Q,I),  *
*  ADB(Q)),Q = 1,5) *
* * * * * * * * * * *
          |
          |
/00217000/|        15
    ----------------
   /                /
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4017      /
----------------
          |
          |
/00217100/|        16
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4018      /
/ FROM THE LIST /
----------------
          |
          |
/00217100/|  NOTE  17
* * * * * * * * * * *
*      LIST =       *
*    ((PASS(Q),     *
*  PCTPAS(Q)),Q =   *
*       1,5)        *
* * * * * * * * * * *
          |
          |
/00217200/|        18
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4019      /
/ FROM THE LIST /
----------------
          |
          |
/00217200/|  NOTE  19
* * * * * * * * * * *
*      LIST =       *
*    ((IFAIL(Q),    *
*   PCTFL(Q)),Q =   *
*       1,5)        *
* * * * * * * * * * *
          |
          |
          +--------------*

          +-----------------*
          |
          |
/00217300/|        20
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4020      /
/ FROM THE LIST /
----------------
          |
          |
/00217300/|  NOTE  21
* * * * * * * * * * *
*      LIST =       *
*  ((ITSETBK(Q,I),  *
*   PCTSB(Q)),Q =   *
*       1,5)        *
* * * * * * * * * * *
          |
          |
/00217400/|        22
    ----------------
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4021      /
/ FROM THE LIST /
----------------
          |
          |
/00217400/|  NOTE  23
* * * * * * * * * * *
*      LIST =       *
*   ((INADIS(Q),    *
*  PCTNDS(Q)),Q =   *
*       1,5)        *
* * * * * * * * * * *
          |
          |
/00217500/|        24
    ----------------
   /                /
  / WRITE TO DEV   /
 /       9        /
/  VIA FORMAT    /
/     4022      /
----------------
          |
          |
/00217600/| 25
    ---------
   * EXIT   *
    ---------

CRT ROUTINE FOR
SCHOOL AND CENTER
TOTALS, ENTRY IS
15000 FOR CENTER

 51.07--->*
   5000   |        26
    ----------------
  / WRITE TO DEV   /
 /       11       /
/  VIA FORMAT    /
/     5002      /
/ FROM THE LIST /
----------------
          |
          |
/00218200/|  NOTE  27
* * * * * * * * * * *
* LIST = SCHOOL(I)  *
* * * * * * * * * * *
          |
          |
/00218500/|        28
    ----------------
  / WRITE TO DEV   /
 /       11       /
/  VIA FORMAT    /
/     5003      /
/ FROM THE LIST /
----------------
          |
          |
/00218500/|  NOTE  29
* * * * * * * * * * *
* LIST = SCHOOL(I)  *
* * * * * * * * * * *
          |
          |
ENTRY CENTER TOTALS
          |
 51.09--->|
  15000   |        30
    ----------------
  / WRITE TO DEV   /
 /       11       /
/  VIA FORMAT    /
/     4009      /
/ FROM THE LIST /
----------------
          |
          |
/00219000/|  NOTE  31
* * * * * * * * * * *
*  LIST = YEAR2D,    *
*  YEAR2D, YEAR2D,   *
*      YEAR2D        *
* * * * * * * * * * *
          |
          |
/00219100/|        32
    ----------------
   /                /
  / WRITE TO DEV   /
 /       11       /
/  VIA FORMAT    /
/    ' 4010      /
----------------
          |
          |
/00219200/|        33
    ----------------
  / WRITE TO DEV   /
 /       11       /
/  VIA FORMAT    /
/     5004      /
/ FROM THE LIST /
----------------
          |
          |
/00219200/|  NOTE  34
* * * * * * * * * * *
* LIST = SCONV(1),  *
*     SCONV(2),     *
*     SCONV(3),     *
*     SCONV(4),     *
*     SCONV(5)      *
* * * * * * * * * * *
          |
          |
          |  /
    -----
   /53.01
```

FIGURE V-17 (CONTINUED)

CHART TITLE - SUBROUTINE CRTOUT(X)

```
52.34--->*
/00219300/I          01
  WRITE TO DEV
  11
  VIA FORMAT
  5005
FROM THE LIST

/00219300/I  NOTE  02
* * * * * * * * * * *
* LIST = SCAP(1),   *
* SCAP(2), SCAP(3), *
* SCAP(4), SCAP(5)  *
* * * * * * * * * * *

/00219400/I          03
  WRITE TO DEV
  11
  VIA FORMAT
  5006
FROM THE LIST

/00219400/I  NOTE  04
* * * * * * * * * * *
*     LIST =        *
*   ((SUTIL(Q),     *
*  PCTUTL(Q)),Q =   *
*       1,5)        *
* * * * * * * * * * *

/00219500/I          05
  WRITE TO DEV
  11
  VIA FORMAT
  5007
FROM THE LIST

/00219500/I  NOTE  06
* * * * * * * * * * *
*     LIST =        *
*   ((SNOSMO(Q),    *
*  PCTNSM(Q)),Q =   *
*       1,5)        *
* * * * * * * * * * *

/00219600/I          07
  WRITE TO DEV
  11
  VIA FORMAT
  14015
FROM THE LIST

/00219600/I  NOTE  08
* * * * * * * * * * *
*     LIST =        *
*   ((SARL(Q),      *
* SARL(Q),Q = 1,5)  *
* * * * * * * * * * *

/00219700/I          09
  WRITE TO DEV
  11
  VIA FORMAT
  5008
FROM THE LIST

/00219700/I  NOTE  10
* * * * * * * * * * *
*     LIST =        *
*   ((SSDAYS(Q),    *
* AOB(Q)),Q = 1,5)  *
* * * * * * * * * * *

/00219800/I          11
  WRITE TO DEV
  11
  VIA FORMAT
  4017

/00219900/I          12
  WRITE TO DEV
  11
  VIA FORMAT
  5009
FROM THE LIST

/00219900/I  NOTE  13
* * * * * * * * * * *
*     LIST =        *
*   ((PASS(Q),      *
*  PCTPAS(Q)),Q =   *
*       1,5)        *
* * * * * * * * * * *

/00220000/I          14
  WRITE TO DEV
  11
  VIA FORMAT
  5010
FROM THE LIST

/00220000/I  NOTE  15
* * * * * * * * * * *
*     LIST =        *
*   ((SFAIL(Q),     *
*  PCTFL(Q)),Q =    *
*       1,5)        *
* * * * * * * * * * *

/00220100/I          16
  WRITE TO DEV
  11
  VIA FORMAT
  5011
FROM THE LIST

/00220100/I  NOTE  17
* * * * * * * * * * *
*     LIST =        *
*   ((SSETBK(Q),    *
*  PCTSB(Q)),Q =    *
*       1,5)        *
* * * * * * * * * * *

/00220200/I          18
  WRITE TO DEV
  11
  VIA FORMAT
  5012
FROM THE LIST

/00220200/I  NOTE  19
* * * * * * * * * * *
*     LIST =        *
*   ((SNADIS(Q),    *
*  PCTNDS(Q)),Q =   *
*       1,5)        *
* * * * * * * * * * *

/00220300/I          20
  WRITE TO DEV
  11
  VIA FORMAT
  4022

/00220400/I 21
  * EXIT *

FORMATS FOR 80 COLUMN
CRT OUTPUT

SCHOOL AND CENTER CRT
FORMATS

/00232500/I 22
  * EXIT *
```

FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
      IMPLICIT INTEGER*2 (A-Z,$)
      REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, COP
      REAL*4 TDMAND, SETBAK
      COMMON /DATA/ COP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK
      INTEGER*4 JNO, PNEC, SCHOOL
      REAL*4 PAOB, PFAIL
      REAL*8 NAME1, NAME2, CATNO1
      COMMON /TEXT/ JNO (176), NAME1 (176), NAME2 (176), CATNO1 (176)
      COMMON /TEXT/  CATNO2 (176), PNEC (176), TYPE (176), PAOB (176)
      COMMON /TEXT/ PFAIL (176), PRTY (176), SCHOOL (176)
      REAL*4 TFAIL, TNADIS
      INTEGER*4 TSDAYS, TCAP, TUTIL
      COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176)
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
      COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
      INTEGER*4 SUTIL, CUTIL, SCAP, CCAP, SSDAYS, CSDAYS, IFAIL, INADIS
      INTEGER*4 PASS, SBKLOG, CBKLOG, SBKL,CBKL
      REAL*4 AOB, PCTUTL, PCTNSH, PCTPAS, PCTFL, PCTSB, PCTNDS, PLEN
      COMMON /T/ SUTIL (5), SCAP (5), SSDAYS (5), SSETBK (5), SNOSHO (5)
      COMMON /T/ CUTIL (5), CCAP (5), CSDAYS (5), CSETBK (5), CNOSHO (5)
      COMMON /T/SBKL(5),SBKLOG(5),SCONV (5), SFAIL (5), SNADIS (5)
      COMMON /T/ CCONV(5),CBKL(5),CBKLOG(5), CFAIL (5), CNADIS (5)
      COMMON /T/ PCTUTL (5), PCTNSH (5), PCTNDS (5), INADIS (5)
      COMMON /T/ AOB (5), PASS (5), PCTPAS (5), PCTFL (5), PCTSB (5)
      COMMON /T/ IFAIL (5), PLEN (3), BKL (5), SPLAT (4), YEAR20
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, BLEN
      COMMON DUMMY, BCONV, BLEN, BCAP, BBCAP, LAST, COURSE
      COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
      COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
      COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCOM
      COMMON TRACEP, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
5001  FORMAT ('1', T23, 'TPF MODEL -- FISCAL YEAR ', I4, ' SUMMARY')
6001  FORMAT (' ', T22, '** TOTALS FOR  FLEET TRAINING CENTER **' /)
6002  FORMAT (' FTC - WORVA  SUMMARY' /)
5002  FORMAT (1X, T33,'SCHOOL CODE  ', A4 /)
5003  FORMAT (' ', A4, 2X, 'SCHOOL SUMMARY' /)
4001  FORMAT ('1', A4, 5X, 2A8, 4X, '--SCHEDULE DATA--   FY   FY+1   FY+
     2   CRS BACKLOG')
4002  FORMAT (' ', T31, 'ANNUAL CONVENINGS', I5, 2I7, 4X, '# STU', I5)
4003  FORMAT (' SCHL JNO  CATALOG NO  NEC    COURSE LENGTH DYS', F5.1,
       2F7.1, 4X, '# WKS', I5)
4004  FORMAT (' ', 2 (A4, 1X), A8, A2, 1X, A4, 4X, 'COURSE CAPACITY',
```

FIGURE V-17 (CONTINUED)

589

```
          317)
4005      FORMAT (' ', T31, 'BUPERS CAPACITY', 317, 3X, 'FY+1 WK', I4)
4006      FORMAT (' P AOB  P'AIL  TYPE   PRTY    ANNUAL DEMAND', 2X, 317,
           3X, 'FY+2 WK', I4)
4007      FORMAT (' ', F5.1, F6.1, '%', 3X, 2 (A2, 5X), 'BUPERS DEMAND',
           2X, 317, 3X, 'OFFSET', I5)
4008      FORMAT (1X, T46, 3 (5X, A2), / 1X, T33, 'QUARTERLY PROJECTIONS')
4009      FORMAT (' ', 12X, '| 1 OR FY', I3, ' | 2 OR FY', I3,
           ' | 3 OR FY', I3, ' | 4 OR FY', I3, ' | ANNUAL TOTAL |')
4010      FORMAT (' ', '------------|------------|------------|',
           '------------|------------|-------------|')
4011      FORMAT (' ', '# CONVENINGS|', I4, ' |', 5X,   ' |', I4, ' |',
           5X,   ' |', I4, ' |', 5X,  ' |', I4, ' |', 5X,   ' |',
           I4, ' |', 5X,   ' |')
4012      FORMAT (' ', 'CRS CAPACITY|', I4, ' |', 5X,   ' |', I4, ' |',
           5X,   ' |', I4, ' |', 5X,  ' |', I4, ' |', 5X,   ' |',
           I4, ' |', 5X,   ' |')
4013      FORMAT (' ', 'UTILIZATION |', I4, ' |', F5.1, '%|', I4, ' |',
           F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
           I4, ' |', F5.1, '%|')
4014      FORMAT (' ', 'NO SHOWS    |', I4, ' |', F5.1, '%|', I4, ' |',
           F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
           I4, ' |', F5.1, '%|')
4015      FORMAT (' ', 'BAKLOG #/WKS|', I4, ' |', I2,' WKS|', I4, ' |',
           I2,' WKS|', I4, ' |', I2,' WKS|',I4, ' |', I2,' WKS|',
           I4, ' |', I2,' WKS|')
4016      FORMAT (' ', 'STU DAYS/AOB|', I4, ' |', F5.1, ' |', I4, ' |',
           F5.1, ' |', I4, ' |', F5.1,' |', I4, ' |', F5.1, ' |',
           I4, ' |', F5.1, ' |')
4017      FORMAT (' ', 12X, '|', 5X, '|', 6X, '|', 5X, '|', 6X, '|', 5X,
           '|', 6X, '|', 5X, '|', 6X, '|', 7X, '|', 6X, '|')
4018      FORMAT (' ', 'TOTAL PASS  |', I4, ' |', F5.1, '%|', I4, ' |',
           F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
           I4, ' |', F5.1, '%|')
4019      FORMAT (' ', 'TOTAL FAIL  |', I4, ' |', F5.1, '%|', I4, ' |',
           F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
           I4, ' |', F5.1, '%|')
4020      FORMAT (' ', 'SETBACK ACAD|', I4, ' |', F5.1, '%|', I4, ' |',
           F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
           I4, ' |', F5.1, '%|')
4021      FORMAT (' ', 'DISENROLL NA|', I4, ' |', F5.1, '%|', I4, ' |',
           F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
           I4, ' |', F5.1, '%|')
4022      FORMAT (' ', '-----------------------------------------',
                       '-----------------------------------------',
           /'/'/' ')
5004      FORMAT (' ', '# CONVENINGS|', I5,  '|', 5X,   ' |', I5,  '|',
           5X,   ' |', I5,  '|', 5X,  ' |', I5,  '|', 5X,   ' |',
```

FIGURE V-17 (CONTINUED)

590

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
          I6, ' |', 5X,   ' |')
5005      FORMAT (' ', 'CRS CAPACITY|', I5,  '|', 5X,   ' |', I5,  '|',
           5X,   ' |', I5,  '|', 5X,  ' |', I5,  '|', 5X,   ' |',
           I6, ' |', 5X,   ' |')
5006      FORMAT (' ', 'UTILIZATION |', I5,  '|', F5.1, '%|', I5,  '|',
           F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
           I6, ' |', F5.1, '%|')
5007      FORMAT (' ', 'NO SHOWS    |', I5,  '|', F5.1, '%|', I5,  '|',
           F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
           I6, ' |', F5.1, '%|')
14015     FORMAT (' ', 'BAKLOG #/WKS|', I5, '|', I2,' WKS|', I5, '|',
           I2,' WKS|', I5, '|', I2,' WKS|',I5, '|', I2,' WKS|',
           I6, ' |', I2,' WKS|')
5008      FORMAT (' ', 'STU DAYS/ADB|', I5,  '|', F6.1,  '|', I5,  '|',
           F6.1,  '|', I5,  '|', F6.1, '|', I5,  '|', F6.1,  '|',
           I6, ' |', F6.1,  '|')
5009      FORMAT (' ', 'TOTAL PASS  |', I5,  '|', F5.1, '%|', I5,  '|',
           F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
           I6, ' |', F5.1, '%|')
5010      FORMAT (' ', 'TOTAL FAIL  |', I5,  '|', F5.1, '%|', I5,  '|',
           F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
           I6, ' |', F5.1, '%|')
5011      FORMAT (' ', 'SETBACK ACAD|', I5,  '|', F5.1, '%|', I5,  '|',
           F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
           I6, ' |', F5.1, '%|')
5012      FORMAT (' ', 'DISENROLL NA|', I5,  '|', F5.1, '%|', I5,  '|',
           F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
           I6, ' |', F5.1, '%|')
```

FIGURE V-17 (CONTINUED)

08/15/75 AUTOFLOW CHART SET - FINAL -KEND- FLOWCHARTS PAGE 57

CHART TITLE - BLOCK DATA

INITIAL VALUES FOR PARAMETERS IN COMMON STORAGE

QUARTERLY AND ANNUAL COURSE TOTALS ACCUMULATION

COMMON FOR COURSE ACTIVE MATRIX ELEMENTS

COMMON FOR SCHOOL AND CENTER TOTALS

08/15/75 AUTOFLOW CHART SET - FINAL -KEND- FLOWCHARTS PAGE 58

CHART TITLE - PROCEDURES

```
 -----------
//00238000//
-----------
     |
     |
     | 01
 ---------
* EXIT   *
 ---------
 RETURN
```

FIGURE V-17 (CONTINUED)

CHART TITLE - NON-PROCEDURAL STATEMENTS

```
IMPLICIT INTEGER*2 (A-Z,$)
REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
REAL*4 TDMAND, SETBAK
COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
COMMON /DATA/ OPFSET (176), DEMAND (3,176), BDMAND (3,176)
COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
COMMON /DATA/ BKLG (176), TDMAND, SETBAK
DATA BKLG/176*0/, CRSRUN /176*0/, BKLGPL, NOSHPL, BPOOL /528*0./
REAL*4 TFAIL, TNADIS
INTEGER*4 TSDAYS, TCAP, TUTIL
COMMON /TOTALS/ TCAP (5, 176),  TUTIL (5, 176),  TSETBK (5, 176)
COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
DATA TCAP /880*0/, TUTIL /880*0/, TSETBK /880*0/
DATA TNADIS /880*0./, TBKLOG /880*0/, TCONV /880*0/
DATA TSDAYS /880*0/, TFAIL /880*0./, TNOSHO /880*0/
REAL*4 ACTCTR
COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)
DATA ACTIVE /250*0/, ACTCTR /250*0./, ACTUTL, ACTNEG /500*0/
INTEGER*4 SUTIL, CUTIL, SCAP, CCAP, SSDAYS, CSDAYS, IFAIL, INADIS
INTEGER*4 PASS, SBKLOG, CBKLOG, SBKL,CBKL
REAL*4 ADB, PCTUTL, PCTNSH, PCTPAS, PCTPL, PCTSB, PCTNDS, PLEN
COMMON /T/ SUTIL (5), SCAP (5), SSDAYS (5), SSETBK (5), SNOSHO (5)
COMMON /T/ CUTIL (5), CCAP (5), CSDAYS (5), CSETBK (5), CNOSHO (5)
COMMON /T/SBKL(5),SBKLOG(5),SCONV (5), SFAIL (5), SNADIS (5)
COMMON /T/ CCONV(5),CBKL(5),CBKLOG(5), CFAIL (5), CNADIS (5)
COMMON /T/ PCTUTL (5), PCTNSH (5), PCTNDS (5), INADIS (5)
COMMON /T/ ADB (5), PASS (5), PCTPAS (5), PCTPL (5), PCTSB (5)
COMMON /T/ IFAIL (5), PLEN (3), BKL (5), SPLAT (4), YEAR2D
DATA SPLAT /' ',3*'  '/
```

FIGURE V-17 (CONTINUED)

08/19/75 INPUT LISTING AUTOFLOW CHART SET - FINAL -KEND- FLOWCHARTS

FORTRAN MODULE (LIST,NAMSQ,SOURCSEQ)

LISTING - PROGRAM KEND

CARD NO **** CONTENTS ****

```
C ************** TPF MODEL - AS OF THURSDAY  JUNE 12, 1975 ***************00000100
C                                                                         00000200
      IMPLICIT INTEGER*2 (A-Z,$)                                          00000300
      REAL*4 SELCRS                                                       00000400
      INTEGER*4 SELSCH                                                    00000500
C                                                                         00000600
C COMMON DATA FOR CLASS DATA MATRIX                                       00000700
C                                                                         00000800
      REAL*4 LEN, NOSHOW, NOSHPL, BXLGPL, BPOOL, NADIS, FAIL, CDP         00000900
      REAL*4 TDMAND, SETBAK                                               00001000
C                                                                         00001100
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)    00001200
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)              00001300
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)          00001400
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)    00001500
      COMMON /DATA/ BAXLOG (176), NOSHOW (176), BXLGPL (176)              00001600
      COMMON /DATA/ BXLG (176), TDMAND, SETBAK                            00001700
C                                                                         00001800
      INTEGER*4 JNO, PNEC, SCHOOL                                         00001900
      REAL*4 PAOB, PFAIL                                                  00002000
      REAL*8 NAME1, NAME2, CATNO1                                         00002100
C                                                                         00002200
      COMMON /TEXT/ JNO (176), NAME1 (176), NAME2 (176), CATNO1 (176)     00002300
      COMMON /TEXT/  CATNO2 (176), PNEC (176), TYPE (176), PAOB (176)     00002400
      COMMON /TEXT/ PFAIL (176), PRTY (176), SCHOOL (176)                 00002500
C                                                                         00002600
C QUARTERLY AND ANNUAL COURSE TOTALS ACCUMULATION                         00002700
C                                                                         00002800
      REAL*4 TFAIL, TNADIS                                                00002900
      INTEGER*4 TSDAYS, TCAP, TUTIL                                       00003000
C                                                                         00003100
      COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176)   00003200
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)   00003300
      COMMON /TOTALS/ TNADIS (5, 176), TBXLOG (5, 176), TCONV (5, 176)    00003400
C                                                                         00003500
C COMMON FOR COURSE ACTIVE MATRIX ELEMENTS                                00003600
C                                                                         00003700
      REAL*4 ACTCTR                                                       00003800
      COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)00003900
C                                                                         00004000
C COMMON FOR COMMUNICATION BETWEEN SUBROUTINES                            00004100
C                                                                         00004200
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE            00004300
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN                      00004400
C                                                                         00004500
      COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE                00004600
      COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5           00004700
      COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5                      00004800
      COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON        00004900
      COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF         00005000
C                                                                         00005100
C DIMENSION AND DATA FOR SPECIAL SCHEDULER ROUTINE                        00005200
C                                                                         00005300
      DIMENSION C (107), ISCHD (4, 50), DIV (13)                          00005400
C                                                                         00005500
      DATA ISCHD/Z1000,3*Z0,Z1000,Z0,Z800,Z0,Z1000,Z100,Z4,Z0,2*Z1000,    00005600
     *Z800,Z1000,Z1004,Z20,Z40,Z200,Z1008,Z100,Z804,Z80,Z810,Z40R,Z81,    00005700
     *Z40,Z1020,Z1040,Z810,Z820,Z841,2*Z104,Z410,Z1084,Z420,Z442,Z210,    00005800
     *Z1084,Z444,Z108,2*Z88R,Z1110,Z844,Z888,Z889,Z222,Z112,Z444,Z1112,   00005900
     *Z488,Z489,Z244,Z1224,Z1124,Z248,Z1244,Z1224,Z1248,Z892,Z924,Z925,   00006000
     *Z492,Z249,Z492,Z1249,Z4A4,Z492,Z4A4,Z4B4,Z452,Z252,Z449,Z1294,      00006100
     *Z1A4A,Z94A,Z452,Z495,Z54A,Z4A5,Z952,Z495,Z554,Z555,Z52A,2*ZAAA,     00006200
     *Z1AA,2*ZAAA,Z1554,Z555,ZAAA,Z1555,ZAAA,Z555,ZAAA,Z1AAA,Z1554,ZAAA, 00006300
     *Z1555,Z1AAA,Z156A,ZAB5,ZAAA,Z156A,Z1AAA,ZAAA,Z1AD5,Z156A,Z1AB4,    00006400
     *Z85A,Z16AD,ZD68,ZB5A,Z685,Z15AD,Z1585,Z15AC,ZDAD,Z1586,ZDB6,Z1B5A, 00006500
     *ZB6D,Z1586,Z16DA,Z18BC,ZDB6,Z1B6D,ZDDB,ZDB6,Z76D,Z16DB,ZDDB,ZEDA, 00006600
     *ZDB7,ZDBB,ZEED,Z1B76,ZB76,Z1DBB,Z1776,Z1DDC,ZEED,Z1BBB,Z1777,ZEEE, 00006700
     *Z7BB,Z1777,ZF7B,Z1BBA,ZEF7,Z1777,ZF7B,Z1BDE,Z7BD,Z1DEF,Z17RE,      00006800
     *Z1EFA,ZEFB,Z1BEF,ZFDF,ZFBE,Z7EF,Z17DF,Z17EF,Z1BF6,ZF7E,Z1FBF,ZFF7, 00006900
     *Z1EFE,ZF7B,Z1F7F,ZFFB,Z1FDE,ZFBF,Z1DFF,ZFFF,ZFFE,Z7FF,2*ZFFF,      00007000
     *Z1EFE,ZFFB,Z1FFF,ZFFF,Z1FFE,Z7FF,Z1FFF,ZFFF,Z1FFE,ZFFF,2*71FFF,    00007100
     *Z1FFE,ZFFF,Z1FFF/                                                   00007200
C                                                                         00007300
      DATA DIV/4096,2048,1024,512,256,128,64,32,16,8,4,2,1/               00007400
C                                                                         00007500
C                                                                         00007600
C                                                                         00007700
C READ RUN PARAMETERS:  'WHERE' IS INPUT DEVICE: 'FRSTYR & LASTYR' ARE    00007800
C FIRST AND LAST FISCAL YEARS OF RUN.                                     00007900
C OVERRIDES - ITEM1(FAIL), ITEM2(NOSHOW), ITEM3(CONV),                    00008000
C ITEM4(LEN), ITEM5(CAP), TDMAND IS DEMAND TREND OVERRIDE - USED YREND    00008100
C Q FLG FOR COURSE SEL, W FLG FOR SCHOOL SEL, I2 FLG FOR LENGTH IN WEEKS00008200
C                                                                         00008300
      READ (5, 1000, END=1998) WHERE, FRSTYR, LASTYR, AOBCON,             00008400
     1 ITEM, TEM, ITEM1, TEM1, ITEM2, TEM2, I, TDMAND, ITEM3, TEM3,       00008500
     2 ITEM4, TEM4, ITEM5, TEM5, I1, SETBAK, Q, SELCRS, W, SELSCH,YRFLG,00008600
     3 WKFLG, TRACEF, PREFLG, RLSFLG, ENDFLG, STAT, $R1, $R2, $R3         00008700
C                                                                         00008800
 1000 FORMAT(I2, 3I4, 1X, 7 (I1, F4.3), 5X, I1, F4.3, 2(I1, A4),1011)     00008900
C                                                                         00009000
C CLAMP SETBACK RATE (AS PCT OF FAILURES). CLAMP AOB CONST IF MISSING     00009100
      IF (I1 .LE. 0) SETBAK = 0.5                                         00009200
      IF (AOBCON .LE. 0) AOBCON = 365                                     00009300
      IF (I .LE. 0) TDMAND = 0                                            00009400
C                                                                         00009500
C SET INIT FLAG - USED TO KEY MODEL PRERUN                                00009600
C CLAMP INIT AS USED TO SET NEGRUN FLAG WHOSE ACTUAL VALUE IS SIGNIFCANT00009700
C                                                                         00009800
      INIT = 1                                                            00009900
      PRINTF = 0                                                          00010000
      NEGFLG = INIT                                                       00010100
C                                                                         00010200
C SET TRACE FLAG - USED FOR DETAILED COURSE RUN PRINTOUT                  00010300
      TRACEF = TRACEF + PREFLG + RLSFLG + ENDFLG                          00010400
C                                                                         00010500
C CLAMP DEVICE TO READER OR DISK FOR DATA INPUT                           00010600
C                                                                         00010700
      IF (WHERE .NE. 10) WHERE = 5                                        00010800
C                                                                         00010900
```

```
C
C FIND NUMBER OF YEARS FOR MODEL RUN - MAX 3 YEARS, PROTECT FOR NEGATIVE
C FRSTYR IS FIRST FY OF RUN, LASTYR IS LAST YR, YEAR IS CURRENT YEAR OF
C RUN, ASSUME 1975 IF MISSING, INVERT IF YRS OUT OF ORDER
C
C TEST FOR MISSING DATA
      YEARUN = 1
      IF (FRSTYR) 26, 26, 27
C
C DATA MISSING-ASSUME 1975 & RUN 1 YEAR
   26 YEAR = 1975
      GO TO 32
C
C HAVE FRST YEAR, TEST FOR LAST  RUN 1 YEAR IF NONE
   27 IF (LASTYR) 31, 31, 28
C
C HERE IF HAVE 1ST & LAST YR, FIND DELTA FOR YEARS RUN
   28 YEARUN = LASTYR - FRSTYR
      IF (YEARUN) 29, 30, 30
C
C NEG DELTA, YEARS OUT OF ORDER - STRAIGHTEN OUT
   29 YEARUN = -YEARUN
      FRSTYR = LASTYR
C
   30 YEARUN = YEARUN + 1
   31 YEAR = FRSTYR
      IF (YEARUN .GT. 3) YEARUN = 3
   32 CONTINUE
C
C
C PRINT CHARACTERISTICS UNIQUE TO THIS MODEL RUN
C
      WRITE (6, 1001)
 1001 FORMAT ('1BEGIN TRAINING PROCESS FLOW MODEL EXECUTION - CHARACTERI
     1STICS FOR THIS RUN ARE:')
C
C WRITE YEARS OF RUN
C
      WRITE (6, 1002) YEAR, YEARUN
 1002 FORMAT (/ ' FIRST FISCAL YEAR OF MODEL RUN', I5, 4X, 'YEARS PROJEC
     *TED', I3)
C
      IF (AOBCON .GT. 300) WRITE (6, 1006) AOBCON
 1006 FORMAT (/' AOB CONSTANT IS', I4, '  WEEKEND DAYS USED IN STUDENT D
     *AYS AND AOB CALC')
C
      IF (AOBCON .LE. 300) WRITE (6, 1007) AOBCON
 1007 FORMAT (/' AOB CONSTANT IS', I4, '  ONLY WORKING DAYS USED IN STUD
     *ENT DAYS AND AOB CALC')
C
      IF (WHERE .EQ. 5) WRITE (6, 1008)
 1008 FORMAT (/ ' COURSE DATA INPUT DEVICE - READER:    STATISTICS INPUT
     * DEVICE - READER')
C
      IF (WHERE .GE. 10) WRITE (6, 1009)
 1009 FORMAT(/' ALL INPUTS THIS RUN FROM UNLOADED DATA BASE')
C
      WRITE (6, 1010) PREFLG, ALSFLG, ENDFLG, YRFLG
 1010 FORMAT(/' TRACE FLAGS THIS RUN:  PRERUN', I2, ': CRS REL', I2, ':
     *CRS END', I2, ': YEAR FORMAT', I2 /)
C
      IF (INIT .GT. 0) WRITE (6, 1012)
 1012 FORMAT ( ' INIT FLAG IS SET - MODEL WILL INIT COURSES IN SESSION A
     *T 1ST QUARTER START' /)
C
      IF (STAT .GT. 0) WRITE (6, 11013)
11013 FORMAT ( ' STAT FLAG IS SET-STATISTICAL FAIL DATA WILL OVERRIDE HI
     *STORICAL RATES' /)
C
C END OF MODEL RUN DATA MESSAGES - OUTLINE ERROR MESSAGES
C
      WRITE (6, 1011)
 1011 FORMAT ( ' -------------------------------------------------------
     *-------------------------')
C
      IF (Q .GT. 0) WRITE (6, 12014) SELCRS
      IF (W .GT. 0) WRITE (6, 12014) SELSCH
12014 FORMAT (' SELECT OPTION- SELECT KEY IS ', A4)
C
C READ COURSE DATA FROM DATA BASE INTERFACE INTO MATRIX
C COURSE MATRIX SET UP FOR MAXIMUM OF 176 INDIVIDUAL COURSES
C CRSES EQUALS NUMBER OF UNIQUE COURSES FOR THIS RUN
C
      CRSES = 0
    1 I = CRSES + 1
C
      READ (WHERE, 10, END=2)   COP (I), CATNO1 (I), CATNO2 (I),
     1 NAME1 (I), NAME2 (I), CONV (1,I), LEN (1,I), CAP (1,I),
     2 BCAP (1,I), DEMAND (1,I), BDMAND (1,I), OFFSET (I), FAIL (I),
     3 NADIS (I), NOSHOW (I), DUMMY     , BAKLOG (I), JNO (I),
     4 CONV (2,I), LEN (2,I), CAP (2,I), BCAP (2,I), DEMAND (2,I),
     5 BDMAND (2,I), CHGWK(1,I), CONV (3,I), LEN (3,I), CAP (3,I),
     6 BCAP (3,I), DEMAND (3,I), BDMAND (3,I), CHGWK (2,I), PAOB (I),
     7 PFAIL (I), PRTY (I), TYPE (I), PNEC (I), SCHOOL (I)
C
   10 FORMAT (A4, A8, A2, 2A8, I3, F3.1, 2I3, 2I5, I2, 4F4.3, I4, A4,2X,
     * /4X,
     1 2 (I3, F3.1, 2I3, 2I5, I3), 2F4.1, 2A2, 2A4)
C
C COURSE / SCHOOL SELECT ROUTINE (Q IS COURSE FLAG, W IS SCHOOL FLAG)
C
      IF (Q) 4, 4, 3
    3 IF (SELCRS .NE. COP (I)) GO TO 1
C
    4 IF (W) 6, 6, 5
    5 IF (SELSCH .NE. SCHOOL (I)) GO TO 1
C
    6 CRSES = CRSES + 1
      IF (CRSES .LT. 176) GO TO 1
C EXIT MODEL IF NO ACTIVE COURSES
    2 IF (CRSES .LE. 0) GO TO 1998
C
```

```
CARD NO      ****                    CONTENTS                    ****

  222              WRITE (6, 12002) CRSES                                00022200
  223       12002 FORMAT (' COURSES ACTIVE THIS RUN', I5)                00022300
  224       C                                                            00022400
  225       C STATISTICAL FAILURE OVERRIDE-IF STAT FLAG ON READ FILE 8 TO OBTAIN   00022500
  226       C NEW FAILURE RATES CALCULATED BY OFFLINE PROGRAM FROM STUDENT CHARACTE-00022600
  227       C RISTICS, OVERRIDING HISTORICAL RATES.  J NO USED AS KEY    00022700
  228       C                                                            00022800
  229              IF (STAT) 3010, 3010, 3000                            00022900
  230        3000 I2 = 0                                                 00023000
  231        3001 READ (8, 3002, END = 3005) WHERE, SELCRS               00023100
  232        3002 FORMAT (A4, F4.3)                                      00023200
  233       C                                                            00023300
  234       C SEE IF COURSE IN OUR FILE                                  00023400
  235              DO 3004 I = 1, CRSES                                  00023500
  236              IF (WHERE .NE. JNO (I)) GO TO 3004                    00023600
  237       C                                                            00023700
  238       C FOUND OUR COURSE - WRITE IN NEW FAIL RATE & BUMP COUNTER   00023800
  239              FAIL (I) = SELCRS                                     00023900
  240              I2 = I2 + 1                                           00024000
  241              GO TO 3001                                            00024100
  242        3004 CONTINUE                                               00024200
  243       C                                                            00024300
  244       C HERE IF COURSE NOT IN FILE - PRINT ERROR MESSAGE           00024400
  245              WRITE (6, 3006) WHERE                                 00024500
  246        3006 FORMAT (' STATISTICS OVERRIDE-COURSE J NO ', A4, ' NOT IN FILE') 00024600
  247              GO TO 3001                                            00024700
  248       C                                                            00024800
  249       C END STATISTICS UPDATE - PRINT TOTAL COURSES FOUND          00024900
  250        3005 WRITE (6, 3007) I2                                     00025000
  251        3007 FORMAT ('STATISTICS UPDATE ROUTINE - ', I3, ' COURSES UPDATED')  00025100
  252        3010 CONTINUE                                               00025200
  253       C                                                            00025300
  254       C                                                            00025400
  255       CC INITIALIZE BACKLOG POOL - BAKLOG IS COURSE BACKLOG IN WEEKS, CONVERT 00025500
  256       C TO STUDENT DEMAND & INIT TO BKLGPL, ZERO BACKLOG IF NEGATIVE INPUT ERR00025600
  257       C INIT BACKLOG POOL TO 8 CONVENINGS IF NO 'A' CONVENINGS.    00025700
  258       C                                                            00025800
  259              DO 720 I = 1, CRSES                                   00025900
  260              IF (BAKLOG (I))711, 720, 712                          00026000
  261         711  BAKLOG (I) = 0                                        00026100
  262              WRITE (6, 11011) COP (I)                              00026200
  263       11011 FORMAT (' NEG BACKLOG COURSE ', A4, '  ZEROED THIS RUN')  00026300
  264              GO TO 720                                             00026400
  265       C                                                            00026500
  266         712  I1 = 1                                                00026600
  267              IF (CONV (I1, I)) 713, 713, 714                       00026700
  268       C NO 'A' CONVENINGS, TEST 8 CONVENINGS                       00026800
  269         713  I1 = 2                                                00026900
  270              IF (CONV (I1, I)) 720, 720, 714                       00027000
  271       C HAVE CONVENINGS, I1 POINTS AT WHICH ONE                    00027100
  272         714  I2 = CAP (I1, I) - BCAP (I1, I)                       00027200
  273              IF (I2) 720, 720, 715                                 00027300
  274         715  SELCRS = 30. / CONV (I1, I)                           00027400
  275              BKLGPL (I) = (BAKLOG (I) / SELCRS) * I2               00027500
  276              BKLG (I) = BKLGPL (I)                                 00027600
  277         720  CONTINUE                                              00027700
  278       C                                                            00027800
  279       C OVERRIDE ROUTINES - FROM PARAMETER CARD                    00027900
  280       C                                                            00028000
  281       C FAIL OVERRIDE - ITEM IS FLAG, TEM IS MULTIPLIER            00028100
  282       C                                                            00028200
  283              IF (ITEM) 91007, 91007, 91001                         00028300
  284       91001 IF (TEM) 91002, 91004, 91004                           00028400
  285       91002 WRITE (6, 91003) TEM                                   00028500
  286       91003 FORMAT (' NEGATIVE OVERRIDE - REQUEST IGNORED. VALUE =', F6.3)  00028600
  287              GO TO 91007                                           00028700
  288       91004 WRITE (6, 91005) TEM                                   00028800
  289       91005 FORMAT (' FAIL OVERRIDE- RATES MULT BY', F6.3, ' THIS RUN ONLY')  00028900
  290              DO 91006 I = 1, CRSES                                 00029000
  291       91006 FAIL (I) = FAIL (I) * TEM                              00029100
  292       C                                                            00029200
  293       C NO SHOW OVERRIDE - FLAG IS ITEM1, MULT IS TEM1             00029300
  294       C                                                            00029400
  295       91007 IF (ITEM1) 91017, 91017, 91011                         00029500
  296       91011 IF (TEM1) 91012, 91014, 91014                          00029600
  297       91012 WRITE (6, 91003) TEM1                                  00029700
  298              GO TO 91017                                           00029800
  299       91014 WRITE (6, 91015) TEM1                                  00029900
  300       91015 FORMAT (' NO SHOW OVERRIDE - RATES MULT BY',F6.3,' THIS RUN ONLY')00030000
  301              DO 91016 I = 1, CRSES                                 00030100
  302       91016 NOSHOW (I) = NOSHOW (I) * TEM1                         00030200
  303       C                                                            00030300
  304       C DEMAND OVERRIDE - FLAG IS ITEM2, MULT IS TEM2              00030400
  305       C                                                            00030500
  306       91017 IF (ITEM2) 91027, 91027, 91021                         00030600
  307       91021 IF (TEM2) 91022, 91024, 91024                          00030700
  308       91022 WRITE (6, 91003) TEM2                                  00030800
  309              GO TO 91027                                           00030900
  310       91024 WRITE (6, 91025) TEM2                                  00031000
  311       91025 FORMAT (' DEMAND OVERRIDE-DEMANDS MULT BY',F6.3,' THIS RUN ONLY')  00031100
  312              DO 91026 I = 1, CRSES                                 00031200
  313              DO 91026 I1 = 1, 3                                    00031300
  314              DEMAND (I1, I) = DEMAND (I1, I) * TEM2 + .5           00031400
  315       91026 BDMAND (I1, I) = BDMAND (I1, I) * TEM2 + .5            00031500
  316       C                                                            00031600
  317       C OVERRIDE ANNUAL CONVENINGS - FLAG IS ITEM3, MULT IS TEM3   00031700
  318       C                                                            00031800
  319       91027 IF (ITEM3) 91037, 91037, 91031                         00031900
  320       91031 IF (TEM3) 91032, 91032, 91034                          00032000
  321       91032 WRITE (6, 91003) TEM3                                  00032100
  322              GO TO 91037                                           00032200
  323       91034 WRITE (6, 91035) TEM3                                  00032300
  324       91035 FORMAT (' CONVENINGS OVERRIDE-FREQ MULT BY',F6.3,' THIS RUN ONLY')00032400
  325              DO 91036 I = 1, CRSES                                 00032500
  326              DO 91036 I1 = 1, 3                                    00032600
  327       91036 CONV (I1, I) = CONV (I1, I) * TEM3 + 0.5               00032700
  328       C                                                            00032800
  329       C CAPACITY OVERRIDE - FLAG IS ITEM5, MULT IS TEM5            00032900
  330       C                                                            00033000
  331       91037 IF (ITEM5) 91057, 91057, 91051                         00033100
  332       91051 IF (TEM5) 91052, 91052, 91054                          00033200
  333       91052 WRITE (6, 91003) TEM5                                  00033300
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
      GO TO 91057                                                       00033400
91054 WRITE (6, 91055) TEM5                                             00033500
91055 FORMAT (' CAPACITY OVERRIDE-CAP MULT BY',F6.3,' THIS RUN ONLY')   00033600
      DO 91056 I = 1, CRSES                                             00033700
      DO 91056 I1 = 1, 3                                                00033800
      CAP (I1, I) = CAP (I1, I) * TEM5 + .5                             00033900
91056 BCAP (I1, I) = BCAP (I1, I) * TEM5 + .5                           00034000
C                                                                       00034100
91057 IF (TDMAND .GT. 0.) WRITE (6,91058) TDMAND                        00034200
91058 FORMAT (' TREND OPTION- DEMANDS MULT BY', F6.3, ' EACH YEAR')     00034300
C                                                                       00034400
C TEST IF LENGTH INPUT IN DAYS, CONVERT TO WEEKS IF IN DAYS FOR EXECUTE 00034500
C                                                                       00034600
      IF (WKFLG) 91061, 91061, 91069                                    00034700
91061 DO 91068 J = 1, 3                                                 00034800
      DO 91068 I = 1, CRSES                                             00034900
      IF (LEN (J,I)) 91068, 91068, 91062                                00035000
91062 ITEM = LEN (J,I) / 7                                              00035100
      TEM = LEN (J,I) - ITEM * 7                                        00035200
      IF (TEM .GT. 5) TEM = 5                                           00035300
      LEN (J,I) = ITEM + TEM / 5.                                       00035400
91068 CONTINUE                                                          00035500
C                                                                       00035600
C LENGTH OVERRIDE - MIN CHANGE - HALF DAYS ONLY                         00035700
C SET TEM TO 1 IF LENGTH INPUT IN DAYS, 10 IF IN WEEKS                  00035800
C                                                                       00035900
91069 IF (ITEM4) 91047, 91047, 91041                                    00036000
91041 IF (TEM4) 91042, 91042, 91044                                     00036100
91042 WRITE (6, 91003) TEM4                                             00036200
      GO TO 91047                                                       00036300
91044 WRITE (6, 91045) TEM4                                             00036400
91045 FORMAT (' LENGTH OVERRIDE-LEN MULT BY',F6.3,' THIS RUN ONLY')     00036500
91049 DO 91046 I = 1, CRSES                                             00036600
      DO 91046 I1 = 1, 3                                                00036700
      IF (LEN (I1,I) .LE. 0) GO TO 91046                                00036800
      ITEM = LEN (I1,I) * TEM4 * 10. + 1                                00036900
      LEN (I1,I) = ITEM / 10.                                           00037000
91046 CONTINUE                                                          00037100
C                                                                       00037200
C TEST COURSE DATA MATRIX FOR INVALID OR NEGATIVE DATA                  00037300
C                                                                       00037400
91047 DO 25 I = 1, CRSES                                                00037500
C                                                                       00037600
      PFAIL (I) = FAIL (I) * 100.                                       00037700
C                                                                       00037800
C CLAMP OFFSET AT 1 YEAR                                                00037900
C                                                                       00038000
      ITEM = OFFSET (I)                                                 00038100
      IF (IABS (ITEM) .LE. 51) GO TO 12                                 00038200
      OFFSET (I) = 0                                                    00038300
      WRITE (6, 10017) COP (I)                                          00038400
10017 FORMAT (' OFFSET GT 1 YEAR FOR COURSE ', A4, '  ZEROED THIS RUN') 00038500
   12 CONTINUE                                                          00038600
C                                                                       00038700
C TEST MULTI DIMENSIONED DATA, TEST ALL FOR I1 = 1, REST ONLY IF CHG WK 00038800
C                                                                       00038900
      DO 25 I1 = 1, 3                                                   00039000
      I2 = I1 - 1                                                       00039100
      IF (I2) 15, 15, 13                                                00039200
C                                                                       00039300
C TEST CHANGE WEEK FOR NEG, SKIP FURTHER TESTS IF NO CHANGE WEEK        00039400
   13 IF (CHGWK (I2, I)) 14, 25, 15                                     00039500
   14 CHGWK (I2, I) = 0                                                 00039600
      WRITE (6, 11035) COP (I)                                          00039700
11035 FORMAT (' NEGATIVE CHANGE WEEK COURSE ',A4,' ZEROED THIS RUN')    00039800
      GO TO 25                                                          00039900
C                                                                       00040000
C TEST CONVENINGS - BYPASS FURTHER TESTS IF 0 CONVENINGS                00040100
   15 IF (CONV (I1, I)) 16, 25, 17                                      00040200
   16 CONV (I1, I) = 0                                                  00040300
      WRITE (6, 10012) COP                                              00040400
10012 FORMAT (' NEGATIVE CONVENINGS FOR COURSE ', A4,' ZEROED THIS RUN')00040500
      GO TO 25                                                          00040600
C                                                                       00040700
   17 IF (LEN (I1, I)) 18, 19, 19                                       00040800
   18 LEN (I1, I) = 0.                                                  00040900
      WRITE (6, 33) COP                                                 00041000
   33 FORMAT (' NEG LENGTH COURSE ', A4, '  ZEROED THIS RUN')           00041100
C                                                                       00041200
   19 IF (CAP (I1, I)) 20, 21, 21                                       00041300
   20 CAP (I1, I) = 0                                                   00041400
      WRITE (6, 34) COP (I)                                             00041500
   34 FORMAT (' NEGATIVE CAPACITY COURSE ', A4, '  ZEROED THIS RUN')    00041600
C                                                                       00041700
   21 IF (BCAP (I1, I)) 22, 23, 23                                      00041800
   22 BCAP (I1, I) = 0                                                  00041900
      WRITE (6, 34) COP (I)                                             00042000
C                                                                       00042100
   23 IF (DEMAND (I1, I)) 24, 701, 701                                  00042200
   24 DEMAND (I1, I) = 0                                                00042300
      WRITE (6, 11018) COP (I)                                          00042400
11018 FORMAT (' NEGATIVE DEMAND FOR COURSE ', A4, '  ZEROED THIS RUN')  00042500
C                                                                       00042600
  701 IF (BDMAND (I1, I)) 702, 25, 703                                  00042700
  702 BDMAND (I1, I) = 0                                                00042800
      WRITE (6, 11018) COP (I)                                          00042900
      GO TO 25                                                          00043000
C                                                                       00043100
  703 IF (BCAP (I1, I))11019, 11019, 25                                 00043200
11019 WRITE (6, 11020) COP (I)                                          00043300
11020 FORMAT (' WARNING-BUPERS DEMAND WITH NO BUPERS SEATS COURSE ', A4)00043400
C                                                                       00043500
   25 CONTINUE                                                          00043600
C                                                                       00043700
C                                                                       00043800
C OUTLINE ERROR MESSAGES                                                00043900
      WRITE (6, 1011)                                                   00044000
C                                                                       00044100
      IF (NEGFLG .GT. 0) WRITE (6, 1013)                                00044200
 1013 FORMAT(/ ' PRERUN FLAG SET- 4TH QTR WILL BE RUN TO INIT CRSES IN S00044300
     *ESSION AT MODEL START',//)                                        00044400
C                                                                       00044500
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
C ****** END MODEL INITITALIZATION - BEGIN ACTIVE MODEL LOOP ******
C
C------------------------------------------------------------------------
C YEAR LOOP - ALLOWS MODEL TO RUN MULTIPLE YEARS, YEAR INPUT ON PARM CRD
C YEARUN HANDED OFF TO OTHER SUBROUTINES TO TEST CHANGE DATE
C------------------------------------------------------------------------
C
      Y = YEARUN
      DO 999 YEARUN = 1, Y
C
C
C QUARTER LOOP - MODEL RUNS ON 4 QUARTERS OF 13 WEEKS
C
C
      DO 900 TEMQ = 1, 4
C
C 10800 IS ENTRY FOR PRERUN END
10800 CONTINUE
      Q = TEMQ
C
C TEST FOR NEGFLAG. OVERRIDE Q IF PRERUN
      IF (NEGFLG .GT. 0) Q = 4
C
C ****** BEGIN LOWEST ORDER (WEEKLY) CYCLING FOR TPF MODEL ******
C LOOP THROUGH ENTIRE COURSE MATRIX ON A WEEKLY BASIS
C
      DO 800 W = 1, 13
C
C
C INITIALIZE 'LAST' TO 1, USED TO FIND LAST USED POSITION IN CRS ACTIVE
C MATRIX BY CRSRLS SUBROUTINE
C
      LAST = 1
      NOW = ((Q - 1) * 13 + W) + (52 * (YEARUN - 1))
C
C LOOP THROUGH COURSE MATRIX
C
      DO 500 I = 1, CRSES
C
C SET UP PROPER COURSE CONVENING, LENGTH, & CAPACITY DATA, DEPENDING ON
C WHETHER OR NOT A COURSE CHANGE DATE HAS PASSED
C USE A CONVENINGS IF NEGFLG (PRERUN) FLAG IS ON
C
      ITEM = 1
      IF (NEGFLG) 11034, 11034, 37
11034 IF (CHGWK (1, I)) 37, 37, 35
C
C HAVE A CHANGE WEEK - CHECK IF EITHER HAS PASSED (1 DOMINANT)
   35 IF (NOW .LT. CHGWK (1, I)) GO TO 37
      ITEM = 2
      IF (CHGWK (2,I)) 36, 36, 39
   39 IF (NOW .LT. CHGWK (2, I)) GO TO 36
C
C CHANGE WEEK 2 HAS PASSED - USE '3' DATA
      ITEM = 3
C IF CONV FREQ LT 8 SET OFFSET TO CHANGE WEEK (TO ASSURE CRS START)
   36 WKOFF = 0
      IF (CONV (ITEM, I) .LT. 8) WKOFF = CHGWK (ITEM-1, I) - 1
      GO TO 38
C
   37 WKOFF = OFFSET (I)
   38 $CONV= CONV (ITEM, I)
      $LEN = LEN (ITEM, I)
      $CAP = CAP (ITEM, I)
      $BCAP= BCAP (ITEM, I)
      COURSE = I + (ITEM * 1000)
C
C TEST FOR CONVENINGS, SKIP IF CONVENINGS LE 0
C
      IF ($CONV) 500, 500, 51
C
C
C SIMULTANEOUS COURSES SET UP FOR THOSE COURSES THAT MEET MORE THAN ONCE
C PER WEEK (50 CONV), OR FOR COURSES WHOSE LENGTH IS SUCH THAT NEW
C SESSIONS CONVENE BEFORE THE PREVIOUS SESSION GRADUATES.
C
C
C SET UP COUNTER 'SIMUL' TO INDICATE NUMBER OF SESSIONS OF THE SAME
C COURSE CONVENING SIMULTANEOUSLY DUE TO CONVENING FREQ GT THAN 1 / WEEK
C MULTIPLE CONVENING SCHEDULES SET UP AS NEEDED IN ARRAY C (I), MAX 10.
C
   51 DIVSR = (($CONV - 1) /50) + 1
      QUOT = $CONV
      SIMUL = DIVSR
C
C NOW BREAK UP COURSE INTO SEPARATE SECTIONS, WITH APPROX SAME NUMBER
C OF INDIVIDUAL CONVENINGS, AND EXACTLY SAME NUMBER TOTAL CONV.
C
      DO 100 II = 1, SIMUL
      IF (DIVSR) 100, 100, 99
   99 C(II) = QUOT / DIVSR
      DIVSR = DIVSR - 1
      QUOT = QUOT - C (II)
  100 CONTINUE
C
C
C CALC OFFSET - OFFSET AVAILABLE AS A MODEL INPUT TO DELAY OR ADVANCE
C A COURSE CONVENING SCHEDULE
C
C TEST FOR OFFSET
      IF (WKOFF) 152, 151, 152
C
C NO OFFSET, SET WEEKS & QUARTERS TO ACTUAL
C
  151 Q1 = Q
      W1 = W
      GO TO 160
C
C HAVE OFFSET, STRAIGHTEN OUT IF NEGATIVE (PRESTART)
C
  152 ITEM = WKOFF
C
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
C MODIFY QUARTERS & WEEKS TO WEEKS
      ITEM1 = (IQ - 1) * 13) + W
C
C SUBTRACT OFFSET
      ITEM1 = ITEM1 - ITEM
C
C SKIP CRSRLS IF 0 OR NEG
      IF (ITEM1) 500, 500, 153
C
C WEEKS POSITIVE, CLAMP TO 52
  153 IF (ITEM1 .GT. 52) ITEM1 = ITEM1 - 52
C
C CONVERT BACK INTO WORKING WEEK FOR SCHEDULE MATRIX
C
      Q1 = (ITEM1 / 13) + 1
      W1 = ITEM1 - ((Q1 - 1) * 13)
C
  160 DO 200 I2 = 1, STMUL
      ITEM = C (I2)
      IF (ITEM) 200, 200, 199
  199 ITEM1 = ITEM (Q1,ITEM) / DIV (W1)
C
      IF (( ITEM1/ 2) * 2) .EQ. ITEM1)  GO TO 200
C
C COURSE SECTION CONVENES THIS WEEK, CALL COURSE RELEASE -TO ACTIVE MTRX
C
      CALL CRSRLS (I)
C
  200 CONTINUE
  500 CONTINUE
C
C SAVE NUMBER OF COURSES RELEASED (OR ACTIVE) THIS WEEK
      NUMRUN = LAST
C
C
C
C **** END COURSE RELEASE - NOW CHECK ACTIVE MATRIX FOR COMPLETIONS ****
C
C
C SEARCH COURSE ACTIVE MATRIX FOR COURSES THAT END THIS WEEK
C ACTIVE (I) CONTAINS RELATIVE COURSE NUMBER, CRS ACTIVE IF GT ZERO
C
C DON'T TERMINATE OR COUNT DOWN COURSES DURING CHRISTMAS RECESS.
C
      IF (Q .EQ. 2 .AND. W .EQ. 13) GO TO 601
      IF (Q .EQ. 3 .AND. W .EQ.  1) GO TO 601
C
C SKIP COURSE END IF NEGFLAG (PRESTART) IS ON
      IF (NEGFLG .GT. 0) GO TO 601
C
      DO 600 I = 1, 250
      IF (ACTIVE (I)) 600, 600, 501
C
C COURSE ACTIVE - SUBTRACT 1 FROM WEEKS TO END & TEST IF COMPLETE
C ACTCTR (I) CONTAINS WEEKS TILL CRS END, COURSE ENDS IF 0 OR NEGATIVE
C
  501 NUMRUN = I
      ACTCTR (I) = ACTCTR (I) - 1.
      IF (ACTCTR (I)) 502, 502, 600
C
C COURSE ENDS THIS WEEK - CALL COURSE END SUBROUTINE
C
  502 CALL CRSEND (I)
C
  600 CONTINUE
  601 CONTINUE
C
C//////////////////////////////////////////////////////////////////////////////
C TRACE PRINT ROUTINE - PRINT ONLY IF PROPER TRACE FLAG IS SET
      IF (TRACEP) 800, 800, 10602
10602 WRITE (6, 11134) W, Q, YEAR, NUMRUN
11134 FORMAT (' TRACE- END OF WEEK    WEEK =', I3, ';   QUARTER =', I2,
     1';   YEAR =', I5, 5X, 'COURSES ACTIVE THIS WEEK =', I4,/)
C//////////////////////////////////////////////////////////////////////////////
C
C ****** END WEEKLY COURSE LOOP - END COURSE START AND COMPLETE ******
C
  800 CONTINUE
C
C TEST IF NEGFLG IS ON, INDICATING THAT QUARTER RUN PRIOR TO THE START
C OF MODEL FISCAL YEAR CYCLING.
C
      IF (NEGFLG) 802, 802, 801
C
C NEGFLG ON, CALL SUBROUTINE PRERUN, CLEAR FLAG, AND BRANCH TO BEGINNING
C OF QUARTER CYCLING
C
  801 CALL PRERUN (2)
      NEGFLG = 0
      GO TO 10800
C
  802 CONTINUE
C
C END QUARTER LOOP - CALL QRTEND SUBROUTINE TO COLLECT QUARTERLY DATA.
C
      CALL QRTEND (Q)
C
  900 CONTINUE
C
C
C END ANNUAL (4 QUARTERS) LOOP - CALL YEAR END SUBROUTINE TO AGGREGATE
C AND PRINT ANNUAL TOTALS.
C
      CALL YEARND (Y)
C
  999 CONTINUE
C
 1998 WRITE (6, 5000) CRSES
      IF (YRPLG .LE. 1) WRITE (11, 5000) CRSES
 5000 FORMAT ('1END OF TPF OUTPUT', I4, ' COURSES PROCESSED', / '/'/'*')
C
      STOP
```

TABLE V-15 (CONTINUED)

```
CARD NO      ****                    CONTENTS                    ****

  670              END                                                    00067000
  671                                                                     00067100
  672              SUBROUTINE PRERUN (X)                                  00067200
  673        C                                                            00067300
  674              IMPLICIT INTEGER*2 (A-Z,$)                             00067400
  675        C                                                            00067500
  676        C COMMON DATA FOR CLASS DATA MATRIX                          00067600
  677        C                                                            00067700
  678              REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP  00067800
  679              REAL*4 TDMAND, SETBAK                                  00067900
  680        C                                                            00068000
  681              COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)  00068100
  682              COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176) 00068200
  683              COMMON /DATA/ OFFSET (176), DEMAND (3,176), RDMAND (3,176)  00068300
  684              COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)  00068400
  685              COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176) 00068500
  686              COMMON /DATA/ BKLG (176), TDMAND, SETBAK               00068600
  687        C                                                            00068700
  688        C COMMON FOR COURSE ACTIVE MATRIX ELEMENTS                   00068800
  689        C                                                            00068900
  690              REAL*4 ACTCTR                                          00069000
  691              COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)00069100
  692        C                                                            00069200
  693        C COMMON FOR COMMUNICATION BETWEEN SUBROUTINES               00069300
  694        C                                                            00069400
  695              INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE  00069500
  696              REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN         00069600
  697        C                                                            00069700
  698              COMMON DUMMY, $CONV, $LEN, $CAP, $$CAP, LAST, COURSE   00069800
  699              COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5  00069900
  700              COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5         00070000
  701              COMMON YEAR, CASES, Q, W, WROFF, INIT, WHERE, NEGFLG, AOBCON  00070100
  702              COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF  00070200
  703        C                                                            00070300
  704        C                                                            00070400
  705        C THIS SUBROUTINE HANDLES THE UNIQUE SITUATION WHERE A NEGATIVE OFFSET  00070500
  706        C HAS BEEN SPECIFIED TO START A COURSE PRIOR TO THE START OF A MODEL RUN00070600
  707        C THIS SITUATION EXISTS FOR COURSES OF RELATIVELY LONG DURATION, AND  00070700
  708        C RELATIVELY FEW CONVENINGS. IT IS USEFUL TO INDICATE TRUE AOB DURING  00070800
  709        C A QUARTER.                                                 00070900
  710        C                                                            00071000
  711        C A CODE OF 0 IS IN (X) IF CALLED BY COURSE RELEASE, 2 IF CALLED BY END.00071100
  712        C                                                            00071200
  713              IF (X) 100, 100, 1000                                  00071300
  714        C                                                            00071400
  715        C CALLED BY RELEASE, TEST IF COURSE SHOULD BE RELEASED. CRITERIA IS THAT00071500
  716        C COURSE LENGTH IS SUCH THAT COURSE WILL STILL BE RUNNING AT MODEL START00071600
  717        C TEST INIT TO SEE IF GENERAL PRERUN OR NEG OFFSET PRERUN    00071700
  718        C                                                            00071800
  719        C                                                            00071900
  720          100 TEM = 14. - W                                          00072000
  721              IF ($LEN .LE. TEM) GO TO 1999                          00072100
  722        C                                                            00072200
  723        C COURSE SHOULD BE RELEASED - SET DUMMY TO 1 FOR RETURN.     00072300
  724        C MODIFY LENGTH TO REFLECT WEEKS REMAINING IN 1ST QUARTER OF COURSE.  00072400
  725        C                                                            00072500
  726              DUMMY = 1                                              00072600
  727              $LEN = $LEN - TEM                                      00072700
  728        C                                                            00072800
  729        C//////////////////////////////////////////////////////////////////////00072900
  730        C TRACE PRINT ROUTINE - PRINT ONLY IF PROPER TRACE FLAG IS SET  00073000
  731              IF (PREFLG) 1999, 1999, 3001                           00073100
  732         3001 WRITE (6, 3000) CDP (COURSE), COURSE, WROFF, $LEN      00073200
  733         3000 FORMAT (' PRERUN CALLED - CDP = ', A4, '  POSITION =', I5,  00073300
  734             1 '  OFFSET =', I4, '  LENGTH IN 1ST QTR =', F4.1)      00073400
  735              GO TO 1999                                             00073500
  736        C//////////////////////////////////////////////////////////////////////00073600
  737        C                                                            00073700
  738        C                                                            00073800
  739        C HERE IF PRERUN CLOSEOUT ROUTINE                            00073900
  740        C                                                            00074000
  741         1000 DO 500 J = 1, 250                                      00074100
  742              IF (ACTIVE (J)) 500, 500, 401                          00074200
  743        C                                                            00074300
  744        C ACTIVE COURSE AT 'J', CALC POSITION IN CLASS DATA MATRIX   00074400
  745        C                                                            00074500
  746          401 ITEM = ACTIVE (J) / 1000                               00074600
  747              ITEM = ACTIVE (J) - ITEM * 1000                        00074700
  748        C                                                            00074800
  749        C FIND LENGTH FOR NEGDAYS CALCULATION.                       00074900
  750        C ACTCTR CONTAINS NUMBER OF WEEKS REMAINING TILL CLOSEOUT OF COURSE.  00075000
  751        C                                                            00075100
  752              $LEN = LEN (1,ITEM) - ACTCTR (J)                       00075200
  753        C                                                            00075300
  754        C//////////////////////////////////////////////////////////////////////00075400
  755        C TRACE PRINT ROUTINE - PRINT ONLY IF PROPER TRACE FLAG IS SET  00075500
  756              IF (PREFLG) 403, 403, 402                              00075600
  757          402 WRITE (6, 4396) CDP (ITEM), ACTCTR (J), $LEN           00075700
  758         4396 FORMAT (' PRESTART CLOSEOUT  CDP =', A4, 3X, 'ACTCTR = ', F5.1,  00075800
  759             1 '    NEGLEN = ', F5.1)                                00075900
  760        C//////////////////////////////////////////////////////////////////////00076000
  761        C                                                            00076100
  762        C TEST AOB CONSTANT FOR WHICH METHOD OF CALC STUDENT DAYS    00076200
  763          403 IF (AOBCON .GT. 300) GO TO 450                         00076300
  764        C                                                            00076400
  765        C HERE IF CLASS DAYS METHOD USED TO CALCULATE AOB.           00076500
  766        C                                                            00076600
  767              ACTNEG (J) = $LEN * ACTUTL (J)  *5.   + .01            00076700
  768              GO TO 500                                              00076800
  769        C                                                            00076900
  770        C HERE IF CALANDER DAYS METHOD USED FOR AOB CALCULATION.     00077000
  771        C                                                            00077100
  772          450 ITEM1 = $LEN - 1                                       00077200
  773              TEM = (ITEM1 * 2) + ($LEN * 5)                         00077300
  774              ACTNEG (J) = TEM * ACTUTL (J) + .01                    00077400
  775        C                                                            00077500
  776          500 CONTINUE                                               00077600
  777        C HERE IF NO MORE ACTIVE COURSES - EXIT SUBROUTINE           00077700
  778        C                                                            00077800
  779              NEGFLG = 0                                             00077900
  780        C                                                            00078000
  781         1999 CONTINUE                                               00078100
```

TABLE V-15 (CONTINUED)

```
CARD NO      ****                    CONTENTS                    ****

 78*       C
 783             RETURN
 784             END
 785
 786             SUBROUTINE CRSRLS (X)
 787       C
 788       C COURSE RELEASE (START) SUBROUTINE - RELEASES COURSE, CALCULATES BUPERS
 789       C SEATS & ATTENDANCE, LOCAL QUOTA CONTROL SCHEDULING, BACKLOG, NOSHOWS,
 790       C SUBSTITUTE QUOTAS, AND ULTIMATE CLASS UTILIZATION, INCLUDING SETBACKS.
 791       C
 792       C COURSE TOTALS - ADDED TO QUARTER TOTALS AS COURSE RELEASES:
 793       C CAPACITY, UTILIZATION, NOSHOWS, BACKLOG, SUBSTITUTE QUOTAS.
 794       C
 795             IMPLICIT INTEGER*2 (A-Z,$)
 796             REAL*4 $DMAND, $$DMND
 797       C
 798       C COMMON DATA FOR CLASS DATA MATRIX
 799       C
 800             REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, COP
 801             REAL*4 TDMAND, SETBAK
 802       C
 803             COMMON /DATA/ COP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
 804             COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
 805             COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
 806             COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
 807             COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
 808             COMMON /DATA/ BKLG (176), TDMAND, SETBAK
 809       C
 810       C QUARTERLY AND ANNUAL COURSE TOTALS ACCUMULATION
 811       C
 812             REAL*4 TFAIL, TNADIS
 813             INTEGER*4 TSDAYS, TCAP, TUTIL
 814       C
 815             COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176)
 816             COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
 817             COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
 818       C
 819       C COMMON FOR COURSE ACTIVE MATRIX ELEMENTS
 820       C
 821             REAL*4 ACTCTR
 822             COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)
 823       C
 824       C COMMON FOR COMMUNICATION BETWEEN SUBROUTINES
 825       C
 826             INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
 827             REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN
 828       C
 829             COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE
 830             COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
 831             COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
 832             COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON
 833             COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
 834       C
 835       C
 836       C SEARCH ACTIVE MATRIX FOR FIRST AVAILABLE SLOT FOR NEW ACTIVE CONVENING.
 837       C ACTIVE (Y) IS ZERO IF NOT IN USE, HAS POINTER TO CRSE MATRIX IF IN USE
 838       C LAST IS POSITION LAST USED TO INSERT A COURSE, START SEARCH FROM THERE
 839       C LAST IS 1 IF THIS IS FIRST COURSE RELEASE FOR THE WEEK.
 840       C
 841       C
 842       C
 843       C TEST IF LOCATION UNUSED IN ACTIVE MATRIX, EXIT IF FREE.
 844       C
 845             DO 10 Y = LAST, 250
 846             IF (ACTIVE (Y)) 50, 50, 10
 847          10 CONTINUE
 848       C
 849       C
 850       C HERE IF TABLE FULL - CANNOT RELEASE COURSE-WRITE ERROR MESSAGE.
 851       C
 852             WRITE (6, 1000)  (COP (X))
 853        1000 FORMAT (1H0, 'COURSE ACTIVE TABLE FULL - COURSE NOT RELEASED',
 854            1 '   COURSE NUMBER IS', A6)
 855       C
 856       C EXIT SUBROUTINE
 857       C
 858             GO TO 999
 859       C
 860       C
 861       C HAVE FOUND OPENING IN TABLE, OPENING IS 'Y', COURSE POINTER IS 'X'.
 862       C
 863       C TEST IF NEGFLG IS ON ( INDICATES MODEL PRESTART CONVENING).
 864       C
 865          50 IF (NEGFLG) 60, 60, 51
 866       C
 867       C HERE IF FLAG IS ON - CALL PRESTART SUBROUTINE TO TEST IF COURSE SHOULD
 868       C BE STARTED.  INIT RETURN CODE (DUMMY) TO 0
 869       C
 870          51 DUMMY = 0
 871             CALL PRERUN (0)
 872       C
 873       C RETURN CODE 1 IF COURSE TO BE STARTED, 0 IF TO SKIP (DUMMY)
 874       C IF COURSE TO BE STARTED, SKIP OVER QUARTERLY TOTAL ACCUMULATE.
 875       C
 876             IF (DUMMY) 999, 999, 63
 877       C
 878       C
 879       C COURSE SHOULD BE RELEASED NORMALLY - ACCUMULATE TOTALS IN QUARTER MTRX
 880       C FOR CONVENINGS AND CAPACITY.
 881       C
 882          60 TCONV (Q, X) = TCONV (Q, X) + 1
 883             TCAP (Q, X) = TCAP (Q, X) + $CAP
 884       C
 885       C LOAD ACTIVE MATRIX & BUMP CRSRUN COUNTER IN COURSE MATRIX, PRESTART
 886       C COURSES ENTER ROUTINE HERE.  SAVE LAST AS NEXT POINTER TO ACTIVE MATRX
 887       C
 888          63 CRSRUN (X) = CRSRUN (X) + 1
 889             ACTIVE (Y) = COURSE
 890             ACTCTR (Y) = $LEN
 891             ACTNEG (Y) = 0
 892             LAST = Y
 893       C
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
C ROUTINE TO ALLOW MULTI YEAR RUN WITH ONLY 1 YEAR SCHEDULE DATA
C CRITERIA, USE REVISED DEMAND IF AVAILABLE, USE CHANGE DATA IF CHG FLG
C
      ITEM = COURSE / 1000
      IF (YEARUN .LT. 2) GO TO 70
      IF (ITEM .GT. 1) GO TO 70
      IF (DEMAND (YEARUN, X)) 70, 70, 65
   65 ITEM = YEARUN
   70 $DMAND = DEMAND (ITEM, X) -BDMAND (ITEM, X)
      IF ($DMAND) 71, 72, 72
   71 $DMAND = 0
   72 $BDMND = BDMAND (ITEM, X)
C
C CHECK IF BUPERS HAS ANY CONTROLLED DEMAND FOR THIS COURSE-ITEM IS NUMB
C CURRENTLY MODEL ALLOWS BUPERS TO OVERBOOK THERE COURSE
C
      ITEM = 0
      IF ($BDMND) 105, 105, 101
C
C BUPERS HAS DEMAND - CALC DEMAND PER CONVENING.
C TEM EQUALS BUPERS STUDENTS (REAL).
C
  101 TEM = 0.
      TEM1 = $CONV
      IF ($CONV) 103, 103, 102
  102 TEM = $BDMND / TEM1
C
C NOW PUT ANY FRACTIONAL DEMAND INTO SAVE POOL, ITEM NOW EQ TO BUPERS
C STUDENTS (INTEGER).
C
  103 ITEM = TEM
      BPOOL (X) = BPOOL (X) + (TEM - ITEM)
C
      IF (BPOOL (X) .LT. 1) GO TO 105
C
C POOL HAS MORE THAN 1 STUDENT, RELEASE & CLEAR POOL OF WHOLE STUDENTS.
C
      ITEM1 = BPOOL (X)
      BPOOL (X) = BPOOL (X) - ITEM1
      ITEM = ITEM + ITEM1
C
C END BUPERS DEMAND, ITEM = NUMBER OF STUDENTS, 0 IF NO BUREAU DEMAND.
C ENTRY IF NO BP DEMAND.
C
C NOW RELEASE LOCAL QUOTA CONTROL DEMAND - SEATS AVAILABLE = CAPACITY
C MINUS BUREAU CONTROLLED SEATS.
C
  105 ITEM1 = $CAP - $BCAP
      IF (ITEM1) 11105, 11106, 11106
11105 ITEM1 = 0
C
C FIND LOCAL DEMAND FOR COURSE - DEMAND = ANNUAL DEMAND / CONVENINGS.
C ADD THIS DEMAND (TEM) TO BKLGPL OR USE AS DEMAND IF IN PRERUN
C IF IN INIT (PRERUN) MODE-SET UP CLASSES BUT DONT ADJUST POOLS
C
11106 TEM = $DMAND / $CONV
      IF (NEGFLG) 107, 107, 106
C
C HERE IF NEG FLG ON- HAVE BUPERS(ITEM), NOW ADD LOCAL - NO SHOW (TEM)
  106 ITEM2 = TEM - (TEM * NOSHOW (X))
      IF (ITEM2 .GT. ITEM1) ITEM2 = ITEM1
      ACTUTL (Y) = ITEM + ITEM2
      GO TO 999
C
C HERE IF NORMAL RELEASE - ADJUST POOLS ETC
C
  107 BKLGPL (X) = BKLGPL (X) + TEM
C
C RELEASE LOCAL STUDENTS - MAKE INIT ASSUMPTION ENTIRE POOL CAN FIT.
C ITEM2 EQUALS LOCAL QUOTA CONTROL STUDENTS.
C
      ITEM2 = BKLGPL (X)
C
C TEST IF POOL GT CAPACITY.
C CLAMP TO CAPACITY IF GREATER.
C ITEM1 IS NOW LOCAL QUOTA CONTROL CAPACITY
C
      IF (ITEM2 .GT. ITEM1) ITEM2 = ITEM1
C
C RELEASE THESE STUDENTS TO COURSE, CLEAR FROM POOL.
      BKLGPL (X) = BKLGPL (X) - ITEM2
C
C NOW TEST FOR LOCAL NO SHOWS
      NOSHPL (X) = NOSHPL (X) + (ITEM2 * NOSHOW (X))
C
C SEE IF GT 1 IN POOL.
      ITEM3 = NOSHPL (X)
      IF (ITEM3) 111, 111, 110
C
C NO SHOWS GT 1, ADD TO TOTALS & TAKE FROM CLASS.
C ITEM3 IS STUDENTS FROM NO SHOW POOL
C
  110 TNOSHO (Q, X) = TNOSHO (Q, X) + ITEM3
      ITEM2 = ITEM2 - ITEM3
      NOSHPL (X) = NOSHPL (X) - ITEM3
  111 CONTINUE
C
C NOW TOTAL STUDENTS & SEE IF FULL CAPACITY.
C ITEM IS BUPERS STUDENTS, ITEM2 IS LOCAL STUDENTS AFTER NO SHOW
C
      ACTUTL (Y) = ITEM + ITEM2
C
      ITEM4 = 0
      IF (ACTUTL (Y) .GE. $CAP) GO TO 114
      ITEM4 = (.1 * $CAP) + .3
C
C CAPACITY NOT FULL - CHECK IF CAN RELEASE THESE FROM BACKLOG POOL
C
      ITEM5 = BKLGPL (X)
      IF (ITEM5) 114, 114, 113
C
C BACKLOG POOL GT 1, RELEASE SUBSTITUTE QUOTAS
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
      C CLAMP STUDENTS TO MAX IN POOL.
      C
        113 IF (ITEM5 .LT. ITEM4) ITEM4 = ITEM5
            BKLGPL (X) = BKLGPL (X) - ITEM4
            ACTUTL (Y) = ACTUTL (Y) + ITEM4
      C
      C
        114 TUTIL (Q,X) = TUTIL (Q,X) + ACTUTL (Y)
        999 CONTINUE
      C
      C//////////////////////////////////////////////////////////////////////////////
      C TRACE PRINT ROUTINE - PRINT ONLY IF PROPER TRACE FLAG IS SET
            IF (RLSFLG) 1010, 1010, 1001
       1001 WRITE (6, 1002) CDP (X), ACTUTL (Y),$LEN, X, LAST
            1 , ITEM, ITEM2, ITEM4, ITEM3
       1002 FORMAT (' RELEASE- CDP= ', A4, ';  STUDENTS= ', I3, ';  LENGTH=',
           1,F5.1,';  CRS SEQ=',I4,';  RUN POSITN=', I4, ';  BUPRS=', I3, ';
           2LOCAL=', I3, ';  SUBQTA=', I2,';  NOSHO=', I3)
      C
            IF (MEGFLG .GT. 0) WRITE (6, 1003) DUMMY
       1003 FORMAT ('+', T132, 'P', I1)
       1010 CONTINUE
      C//////////////////////////////////////////////////////////////////////////////
      C
            RETURN
            END

            SUBROUTINE CRSEND (X)
      C
      C SUBROUTINE CRSEND CALCULATES PASS/FAIL/SETBACK, NACDISENROLLMENT,
      C AND CALCULATES UTILIZATION, TOTALS FOR QUARTERLY REPORT
      C
            IMPLICIT INTEGER*2 (A-Z,$)
      C
      C COMMON DATA FOR CLASS DATA MATRIX
      C
            REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
            REAL*4 TDMAND, SETBAK
      C
            COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
            COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
            COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
            COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
            COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
            COMMON /DATA/ BKLG (176), TDMAND, SETBAK
      C
      C QUARTERLY AND ANNUAL COURSE TOTALS ACCUMULATION
      C
            REAL*4 TFAIL, TNADIS
            INTEGER*4 TSDAYS, TCAP, TUTIL
      C
            COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176)
            COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
            COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
      C
      C COMMON FOR COURSE ACTIVE MATRIX ELEMENTS
      C
            REAL*4 ACTCTR
            COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)
      C
      C COMMON FOR COMMUNICATION BETWEEN SUBROUTINES
      C
            INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
            REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN
      C
            COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE
            COMMON $P1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
            COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
            COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, MEGFLG, AORCON
            COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
      C
      C
      C
            I2 = 0
            ITEM2 = 0
            J = ACTIVE (X)
      C
      C CALLING PARAMETER 'X' IS SUBSCRIPT OF COURSE IN ACTIVE MATRIX.
      C ACTIVE (X) CONTAINS RELATIVE COURSE NUMBER IN COURSE MATRIX, HAS 1000
      C ADDED IF '1' SCHED, 2000 IF '2' SCHED, 3000 IF '3' SCHED
      C
            ITEM5 = J / 1000
            J = J - ITEM5 * 1000
      C
            $LEN = LEN (ITEM5, J)
      C
      C CALCULATE FAILURES - FAILURE RATE FROM REGRESSION CONSTANTS
      C NOTE - J POINTS TO COURSE MATRIX, X POINTS TO ACTIVE MATRIX
      C
            TEM = FAIL (J) * ACTUTL (X)
            TFAIL (Q, J) = TFAIL (Q, J) + TEM
      C
      C CALCULATE NON-ACADEMIC DISENROLLMENT - RATE FROM DATA BASE
      C
            TEM1 = NADIS (J) * ACTUTL (X)
            TNADIS (Q, J) = TNADIS (Q, J) + TEM1
      C
      C TEST FOR SETBACK POSSIBILITY
      C CRITERIA FOR SETBACK: COURSE GT 2 WEEKS, MULTIPLE CONVENINGS, FAILURE
      C RATE TO PRODUCE SETBACKS.
      C
            IF ($LEN .LT. 2.0) GO TO 100
            IF (CRSRUN (J) .LT. 2) GO TO 100
            TEM2 = TEM * SETBAK + .5
            IF (TEM2 .LT. 1.) GO TO 100
      C
      C HERE IF SETBACKS MAY OCCUR
      C
            GO TO 200
      C
      C TOTAL COURSE AND CLOSEOUT ACTIVE MATRIX FOR COURSE
      C
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
  100 ACTIVE (X) = 0
C
C CALCULATE TOTAL STUDENT DAYS FOR AOB CALCULATION.
C ACTNEG CONTAINS DAYS CHARGED TO OTHER QUARTER OR OTHER CONVENINGS.
C
C TEST WHICH METHOD OF AOB CALCULATION TO BE USED.
      IF (AOBCON .GT. 300) GO TO 430
C
C HERE IF CLASS DAYS METHOD TO BE USED
      TSDAYS (Q, J) = TSDAYS (Q, J) + ((BLEN * 5. * ACTUTL (X))
     1 + .01) - ACTNEG (X)
      GO TO 440
C
C HERE IF YEARLY METHOD TO BE USED.
C
  430 ITEM1 = BLEN - 1
      TEM = (ITEM1 * 2) + (BLEN * 5.)
      TSDAYS (Q, J) = TSDAYS (Q, J) +(TEM * ACTUTL (X) +.01)
     1 - ACTNEG (X)
  440 CONTINUE
C
C CLOSE OUT COURSE. RESET ACTIVE & RUN FLAGS.
C
      ACTIVE (X) = 0
      CRSRUN (J) = CRSRUN (J) - 1
      IF (CRSRUN (J) .LT. 0) CRSRUN (J) = 0
      GO TO 999
C
C
C ******************** SETBACK ROUTINE ********************
C HAVE SETBACKS - SEARCH OUT PROPER CONVENING TO SET BACK TO.
C
C INIT TEST VALUES
  200 ITEM5 = 1
      ITEM = ACTIVE (X)
      TEM5 = 999.
      ITEM2 = 0
      ITEM3 = 0
C
C
      DO 120 I2 = 1, 250
      ITEM1 = ACTIVE (I2)
      IF (ITEM1) 120, 120, 101
C HAVE ACTIVE COURSE - TEST IF OURS; DUMMY=SCHED, ITEM1=CRSNO
  101 DUMMY = ITEM1 / 1000
      ITEM1 = ITEM1 - DUMMY * 1000
      IF (ITEM1 .NE. J) GO TO 120
C HAVE OUR COURSE NUMBER-TEST IF ONE WE ARE ENDING
      IF (I2 .EQ. X) GO TO 118
C
C FOUND ONE OF OUR COURSES-TEST IF EARLIEST CONVENING.
      IF (ACTCTR (I2) .GT. TEM5) GO TO 118
C
C EARLIEST SO FAR, SAVE WEEKS & POSITION
      TEM5 = ACTCTR (I2)
      ITEM3 = I2
C
C SEE IF WE FOUND ALL CONVENINGS. BUMP COUNTER & TEST.
C
  118 ITEM2 = ITEM2 + 1
      IF (ITEM2 .EQ. CRSRUN (J)) GO TO 121
  120 CONTINUE
C
C HERE IF DONE COURSE SEARCH OF ACTIVE MATRIX. ITEM3 IS 0 IF NO HIT.
C STATEMENT 100 IS BACKWARD BRANCH TO CLOSEOUT ROUTINE
C
  121 IF (ITEM3) 100, 100, 122
C
C FIND COURSE AND MODIFY TABLES TO REPRESENT SETBACKS
C
  122 ITEM = ACTIVE (ITEM3)
      DUMMY = ITEM / 1000
      ITEM = ITEM - DUMMY * 1000
      TEM4 = LEN (DUMMY, ITEM)
C
      ITEM2 = TEM2
      ACTUTL (ITEM3) = ACTUTL (ITEM3) + ITEM2
      TSETBK (Q, ITEM) = TSETBK (Q, ITEM) + ITEM2
C
C
C NOW CALCULATE NEGATIVE STUDENT DAYS - TO COMPENSATE COURSE STUD LEFT.
C ITEM2 IS SETBACKS, LEN IS FIRST COURSE LENGTH, TEM4 IS SET BACK TO CRS
C LENGTH, TEM5 IS WKS LEFT IN SET BACK TO COURSE.
C
C NOW LET TEM5 EQUAL WEEKS STUDENT HAS BEEN IN SET BACK TO COURSE.
      TEM5 = TEM4 - TEM5
C
C TEST AOB MULT FOR METHOD OF AOB CALCULATION.
      IF (AOBCON .GT. 300) GO TO 130
C
C AOBCON LT 300, USE STUDENT DAYS METHOD.
      ACTNEG (X) = ACTNEG (X) +((TEM5 * 5. * TEM2) + .01)
      GO TO 100
C
C YEARLY AOB METHOD USED - CONVERT WEEKS TO INTEGER.
C STUDENT BEGINS SETBACK TO CLASS MONDAY, WEEKEND CHARGED TO OLD COURSE
C CALC TOTAL STUD DAYS, INCLUDING WEEKENDS - DUMMY IS INTEGER.
C
  130 DUMMY = TEM5
      TEM5 = (DUMMY * 2) +(TEM5 * 5.)
      ACTNEG (X) = ACTNEG (X) +(TEM5 * TEM2 + .01)
      GO TO 100
C
C
  999 CONTINUE
C
C////////////////////////////////////////////////////////////////////////
C
      IF (ENDFLG) 1010, 1010, 1001
 1001 WRITE (6, 1002) COP (J), ACTUTL (X),BLEN, J, X
 1002 FORMAT (' CRS END- COP= ', A4, '; STUDENTS= ', I3, '; LENGTH='
     1,F5.1,'; CRS SEQ=',I4,'; RUN POSITN=', I4)
```

TABLE V-15 (CONTINUED)

```
CARD NO      ****                    CONTENTS                    ****

 1230        C                                                          00123000
 1231              IF (ITEM2 .GT. 0) WRITE (6, 1005) ITEM2, ITEM3       00123100
 1232         1005 FORMAT ('+', T105, I2, ' STONTS SET TO CRS ', I4)    00123200
 1233         1010 CONTINUE                                             00123300
 1234        C                                                          00123400
 1235        C//////////////////////////////////////////////////////////00123500
 1236        C                                                          00123600
 1237              RETURN                                               00123700
 1238              END                                                  00123800
 1239                                                                   00123900
 1240              SUBROUTINE QRTEND (X)                                00124000
 1241        C                                                          00124100
 1242        C QUARTER END ROUTINE TOTALS STUDENT DAYS OF ALL CONVENINGS IN SESSION. 00124200
 1243        C AND ADDS THEM TO COURSE TOTALS.  THESE DAYS ARE ALSO ENTERED AS 00124300
 1244        C 'NEGATIVE' DAYS IN ACTIVE MATRIX TO PREVENT THEM FROM BEING COUNTED IN00124400
 1245        C  NEXT QUARTERS TOTALS.                                   00124500
 1246        C ROUTINE CHECKS AOBCON FOR WHICH METHOD OF TOTALLING AOB IS USED. 00124600
 1247        C                                                          00124700
 1248        C                                                          00124800
 1249              IMPLICIT INTEGER*2 (A-Z,$)                           00124900
 1250        C                                                          00125000
 1251        C COMMON DATA FOR CLASS DATA MATRIX                        00125100
 1252        C                                                          00125200
 1253              REAL*4 LEN, NOSHOW, NOSHPL, $KLGPL, $POOL, NADIS, FAIL, COP 00125300
 1254              REAL*4 TDMAND, SETBAK                                00125400
 1255        C                                                          00125500
 1256              COMMON /DATA/ COP (176), CRSRUN (176), CONV (3,176), LEN (3,176) 00125600
 1257              COMMON /DATA/ CAP (3,176), $CAP (3,176), CHGWK (2,176) 00125700
 1258              COMMON /DATA/ OFFSET (176), DEMAND (3,176), $DMAND (3,176) 00125800
 1259              COMMON /DATA/ $POOL (176), FAIL (176), NADIS (176), NOSHPL (176) 00125900
 1260              COMMON /DATA/ $AKLOG (176), NOSHOW (176), $KLGPL (176) 00126000
 1261              COMMON /DATA/ $KLG (176), TDMAND, SETBAK             00126100
 1262        C                                                          00126200
 1263        C QUARTERLY AND ANNUAL COURSE TOTALS ACCUMULATION          00126300
 1264        C                                                          00126400
 1265              REAL*4 TFAIL, TNADIS                                 00126500
 1266              INTEGER*4 TSDAYS, TCAP, TUTIL                        00126600
 1267        C                                                          00126700
 1268              COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176) 00126800
 1269              COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176) 00126900
 1270              COMMON /TOTALS/ TNADIS (5, 176), T$KLOG (5, 176), TCONV (5, 176) 00127000
 1271        C                                                          00127100
 1272        C COMMON FOR COURSE ACTIVE MATRIX ELEMENTS                 00127200
 1273        C                                                          00127300
 1274              REAL*4 ACTCTR                                        00127400
 1275        C                                                          00127500
 1276              COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)00127600
 1277        C                                                          00127700
 1278        C COMMON FOR COMMUNICATION BETWEEN SUBROUTINES             00127800
 1279        C                                                          00127900
 1280              INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE 00128000
 1281              REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN       00128100
 1282        C                                                          00128200
 1283              COMMON DUMMY, $CONV, $LEN, $CAP, $$CAP, LAST, COURSE 00128300
 1284              COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5 00128400
 1285              COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5       00128500
 1286              COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON 00128600
 1287              COMMON TRACEP, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF 00128700
 1288        C                                                          00128800
 1289        C                                                          00128900
 1290        C                                                          00129000
 1291        C                                                          00129100
 1292        C                                                          00129200
 1293        C TEST ACTIVE MATRIX FOR ACTIVE COURSES.                   00129300
 1294        C                                                          00129400
 1295              DO 500 J = 1, 250                                    00129500
 1296              IF (ACTIVE (J)) 500, 500, 401                        00129600
 1297        C                                                          00129700
 1298        C ACTIVE FILE HAS A COURSE ACTIVE AT 'J'.                  00129800
 1299        C GET POINTER INTO COURSE MATRIX, .                        00129900
 1300        C                                                          00130000
 1301          401 ITEM = ACTIVE (J)                                    00130100
 1302              DUMMY = ITEM / 1000                                  00130200
 1303              ITEM = ITEM - DUMMY * 1000                           00130300
 1304              $LEN = LEN (DUMMY, ITEM)                             00130400
 1305        C                                                          00130500
 1306        C TEST WHICH METHOD OF AOB CALCULATION TO BE USED.         00130600
 1307              IF (AOBCON .GT. 300) GO TO 430                       00130700
 1308        C                                                          00130800
 1309        C C HERE IF CLASS DAYS METHOD TO BE USED.                  00130900
 1310              TEM = ($LEN - ACTCTR (J)) * ACTUTL (J)      *5.      00131000
 1311              GO TO 440                                            00131100
 1312        C                                                          00131200
 1313        C HERE IF YEARLY AOB METHOD TO BE USED                     00131300
 1314        C CHARGE THIS WEEKEND TO THE COURSE BEING CLOSED OUT       00131400
 1315        C                                                          00131500
 1316          430 TEM1 =$LEN - ACTCTR (J)                              00131600
 1317              ITEM1 = TEM1                                         00131700
 1318              TEM = (ITEM1 * 2) + (ITEM1 * 5)                      00131800
 1319              TEM = TEM * ACTUTL (J)                               00131900
 1320          440 CONTINUE                                             00132000
 1321        C                                                          00132100
 1322        C ADD TOTAL STUDENT DAYS TO COURSE & ACTIVE MATRICES       00132200
 1323        C                                                          00132300
 1324              TEM = TEM + .01                                      00132400
 1325              ACTNEG (J) = ACTNEG (J) + TEM                        00132500
 1326              TSDAYS (Q, ITEM) = TSDAYS (Q, ITEM) + TEM            00132600
 1327          500 CONTINUE                                             00132700
 1328        C                                                          00132800
 1329        C                                                          00132900
 1330        C COPY CURRENT BACKLOG POOL FIGURES INTO COURSE TOTALS     00133000
 1331        C                                                          00133100
 1332              DO 8111 I = 1, CRSES                                 00133200
 1333         8111 T$KLOG (Q,I) = $KLGPL (I)                            00133300
 1334        C                                                          00133400
 1335        C COPY FRACTIONAL FAIL & NADIS INTO NEXT QUARTER TOTALS. SKIP IF 4TH QTR00133500
 1336              IF (Q .GE. 4) GO TO 8113                             00133600
 1337              J = Q + 1                                            00133700
 1338              DO 8112 I = 1, CRSES                                 00133800
 1339              ITEM = TFAIL (Q,I)                                   00133900
 1340              TFAIL (J,I) = TFAIL (Q,I) - ITEM                     00134000
 1341              ITEM = TNADIS (Q,I)                                  00134100
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
 8112 TNADIS (J,I) = TNADIS (Q,I) - ITEM                                00134200
 8113 CONTINUE                                                          00134300
C                                                                       00134400
C///////////////////////////////////////////////////////////////////////00134500
      IF (TRACEP .GT. 0) WRITE (6, 1000) X                              00134600
 1000 FORMAT (' QUARTER END ROUTINE FOR QUARTER ', I1)                  00134700
C///////////////////////////////////////////////////////////////////////00134800
C                                                                       00134900
      RETURN                                                            00135000
      END                                                               00135100
                                                                        00135200
      SUBROUTINE YEARND (Y)                                             00135300
C                                                                       00135400
C THIS SUBROUTINE CALCULATES ANNUAL TOTALS AND CALLS THE PROPER SUBROUT-00135500
C INE FOR PRINTOUT OR CRT OUTPUT.  IF A TRACE FLAG IS  ON  THIS SUBR IS 00135600
C BYPASSED AND A TRACE ONLY SUBROUTINE IS CALLED.                       00135700
C                                                                       00135800
C TOTALS ARE AGGREGATED BY COURSE, SCHOOL, AND CENTER ON AN ANNUAL BASIS00135900
C WITH NO CARRY OVER TOTALS FROM YEAR TO YEAR.                          00136000
C                                                                       00136100
C                                                                       00136200
      IMPLICIT INTEGER*2 (A-Z,$)                                        00136300
C                                                                       00136400
C COMMON FOR CLASS DATA MATRIX                                          00136500
C                                                                       00136600
      REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP       00136700
      REAL*4 TDMAND, SETBAK                                             00136800
C                                                                       00136900
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)  00137000
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)            00137100
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)        00137200
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)  00137300
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)            00137400
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK                          00137500
C                                                                       00137600
      INTEGER*4 JNO, PNEC, SCHOOL                                       00137700
      REAL*4 PAOB, PFAIL                                                00137800
      REAL*8 NAME1, NAME2, CATNO1                                       00137900
C                                                                       00138000
      COMMON /TEXT/ JNO (176), NAME1 (176), NAME2 (176), CATNO1 (176)   00138100
      COMMON /TEXT/  CATNO2 (176), PNEC (176), TYPE (176), PAOB (176)   00138200
      COMMON /TEXT/ PFAIL (176), PRTY (176), SCHOOL (176)               00138300
C                                                                       00138400
      REAL*4 TFAIL, TNADIS                                              00138500
      INTEGER*4 TSDAYS, TCAP, TUTIL                                     00138600
C                                                                       00138700
      COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176) 00138800
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176) 00138900
      COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)  00139000
C                                                                       00139100
C COMMON FOR SCHOOL AND CENTER TOTALS                                   00139200
C                                                                       00139300
      INTEGER*4 SUTIL, CUTIL, SCAP, CCAP, SSDAYS, CSDAYS, IFAIL, INADIS 00139400
      INTEGER*4 PASS, SBKLOG, CBKLOG, SBKL,CBKL                         00139500
      REAL*4 AOB, PCTUTL, PCTNSH, PCTPAS, PCTFL, PCTSB, PCTNDS, PLEN    00139600
C                                                                       00139700
      COMMON /T/ SUTIL (5), SCAP (5), SSDAYS (5), SSETBK (5), SNOSHO (5)00139800
      COMMON /T/ CUTIL (5), CCAP (5), CSDAYS (5), CSETBK (5), CNOSHO (5)00139900
      COMMON /T/SBKL(5),SBKLOG(5),SCONV (5), SFAIL (5), SNADIS (5)      00140000
      COMMON /T/ CCONV(5),CBKL(5),CBKLOG(5), CFAIL (5), CNADIS (5)      00140100
      COMMON /T/ PCTUTL (5), PCTNSH (5), PCTNDS (5), INADIS (5)         00140200
      COMMON /T/ AOB (5), PASS (5), PCTPAS (5), PCTFL (5), PCTSB (5)    00140300
      COMMON /T/ IFAIL (5), PLEN (3), BKL (5), SPLAT (4), YEAR2D        00140400
C                                                                       00140500
C COMMON FOR COMMUNICATION BETWEEN SUBROUTINES                          00140600
C                                                                       00140700
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE          00140800
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN                    00140900
C                                                                       00141000
      COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE              00141100
      COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5         00141200
      COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5                    00141300
      COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON      00141400
      COMMON TRACEP, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF       00141500
C                                                                       00141600
C                                                                       00141700
C YRFLG 0 FOR CRT OUTPUT, 1 FOR BOTH, 2 FOR PRINT ONLY, 9 FOR TRACE ONLY00141800
      IF (YRFLG .GE. 9) GO TO 29999                                     00141900
C                                                                       00142000
C SET SPLAT TO KEY YEAR OF RUN                                          00142100
      IF (YEARUN .LE. 1) GO TO 81001                                    00142200
      ITEM = YEARUN - 1                                                 00142300
      SPLAT (YEARUN) = SPLAT (ITEM)                                     00142400
      SPLAT (ITEM) = SPLAT (4)                                          00142500
81001 CONTINUE                                                          00142600
C                                                                       00142700
C INIT & CLAMP RUN CONSTANTS                                            00142800
      ITEM5 = 0                                                         00142900
      IF (AOBCON .LE. 0) AOBCON = 250                                   00143000
      YEAR2D = YEAR - 1900                                              00143100
C                                                                       00143200
C INITIALIZE CENTER TOTALS                                              00143300
C                                                                       00143400
      DO 1 Q = 1, 5                                                     00143500
      CUTIL (Q) = 0                                                     00143600
      CCAP  (Q) = 0                                                     00143700
      CSETBK (Q) = 0                                                    00143800
      CSDAYS (Q) = 0                                                    00143900
      CNOSHO (Q) = 0                                                    00144000
      CCONV  (Q) = 0                                                    00144100
      CFAIL  (Q) = 0                                                    00144200
      CNADIS (Q) = 0                                                    00144300
      CBKL (Q) = 0                                                      00144400
      CBKLOG (Q) = 0                                                    00144500
C                                                                       00144600
      SUTIL (Q) = 0                                                     00144700
      SCAP  (Q) = 0                                                     00144800
      SSETBK(Q) = 0                                                     00144900
      SSDAYS(Q) = 0                                                     00145000
      SNOSHO(Q) = 0                                                     00145100
      SCONV (Q) = 0                                                     00145200
      SFAIL (Q) = 0                                                     00145300
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
      SNADIS(Q) = 0                                                     00145400
      SBKL (Q) = 0                                                      00145500
    1 SBKLOG (Q) = 0                                                    00145600
C                                                                       00145700
C                                                                       00145800
C CALCULATE TOTALS & PRINT FOR EACH COURSE                               00145900
      DO 4444 I = 1, CRSES                                              00146000
C                                                                       00146100
C TEST FOR SCHOOL CODE CHANGE                                           00146200
C DONT PRINT SCHOOL SUMMARY IF THIS IS FIRST PASS                       00146300
C                                                                       00146400
      IF (I .EQ. 1) GO TO 4                                             00146500
      IF (SCHOOL (I) .EQ. SCHOOL (I-1)) GO TO 4                         00146600
C                                                                       00146700
C NEW SCHOOL CODE, CALC OLD SCHOOL TOTALS & PRINT.                      00146800
      GO TO 500                                                         00146900
C                                                                       00147000
C RETURN FROM PRINT SCHOOL TOTALS - NOW CALC TOTALS FOR CLASS           00147100
C ITEMS IS COUNTER FOR COURSE ACTUALLY CALCULATING                      00147200
C                                                                       00147300
    4 ITEMS = ITEMS + 1                                                 00147400
C                                                                       00147500
C USE 5TH POSITION IN TOTALS MATRIX TO AGGREGATE TOTALS - INIT TO ZERO  00147600
      TUTIL (5, I) = 0                                                  00147700
      TCAP (5, I) = 0                                                   00147800
      TSETBK (5, I) = 0                                                 00147900
      TSDAYS (5, I) = 0                                                 00148000
      TNOSHO (5, I) = 0                                                 00148100
      TCONV (5, I) = 0                                                  00148200
C                                                                       00148300
C INIT INTEGER FAIL & NON-ACADEMIC DROPOUT TOTALS TO ZERO               00148400
C                                                                       00148500
      IFAIL (5) = 0                                                     00148600
      INADIS (5) = 0                                                    00148700
C                                                                       00148800
C                                                                       00148900
C NOW TOTAL ANNUAL FIGURES                                              00149000
C                                                                       00149100
      DO 5 Q = 1, 4                                                     00149200
C                                                                       00149300
      TUTIL(5,I) = TUTIL(5,I) + TUTIL(Q,I)                              00149400
      TCAP(5,I) = TCAP(5,I) + TCAP(Q,I)                                 00149500
      TSETBK(5,I) = TSETBK(5,I) + TSETBK(Q,I)                           00149600
      TSDAYS(5,I) = TSDAYS(5,I) + TSDAYS(Q,I)                           00149700
      TNOSHO (5, I) = TNOSHO (5, I) + TNOSHO (Q, I)                     00149800
      TCONV(5,I) = TCONV(5,I) + TCONV(Q,I)                              00149900
C                                                                       00150000
C CONVERT TO INTEGER, ROUND, & TOTAL FAILURES & NON-ACADEMIC DISENROLLS 00150100
C                                                                       00150200
      IFAIL (Q) = 0                                                     00150300
      IFAIL (Q) = TFAIL (Q, I) + .5                                     00150400
      IFAIL (5) = IFAIL (5) + IFAIL (Q)                                 00150500
C                                                                       00150600
      INADIS (Q) = 0                                                    00150700
      INADIS (Q) = TNADIS (Q, I) + .5                                   00150800
      INADIS (5) = INADIS (5) + INADIS (Q)                              00150900
C                                                                       00151000
C TOTAL DATA FOR INDIVIDUAL SCHOOL TOTALS                               00151100
C                                                                       00151200
      SUTIL (Q) = SUTIL (Q) + TUTIL (Q, I)                              00151300
      SCAP (Q) = SCAP (Q) + TCAP (Q, I)                                 00151400
      SSETBK (Q) = SSETBK (Q) + TSETBK (Q, I)                           00151500
      SSDAYS (Q) = SSDAYS (Q) + TSDAYS (Q, I)                           00151600
      SNOSHO (Q) = SNOSHO (Q) + TNOSHO (Q, I)                           00151700
      SCONV (Q) = SCONV (Q) + TCONV (Q, I)                              00151800
      SFAIL (Q) = SFAIL (Q) + IFAIL (Q)                                 00151900
      SNADIS (Q) = SNADIS (Q) + TNADIS (Q, I)                           00152000
C                                                                       00152100
C TOTAL DATA FOR FLEET TRAINING CENTER TOTALS                           00152200
C                                                                       00152300
      CUTIL (Q) = CUTIL (Q) + TUTIL (Q, I)                              00152400
      CCAP (Q) = CCAP (Q) + TCAP (Q, I)                                 00152500
      CSETBK (Q) = CSETBK (Q) + TSETBK (Q, I)                           00152600
      CSDAYS (Q) = CSDAYS (Q) + TSDAYS (Q, I)                           00152700
      CNOSHO (Q) = CNOSHO (Q) + TNOSHO (Q, I)                           00152800
      CCONV (Q) = CCONV (Q) + TCONV (Q, I)                              00152900
      CFAIL (Q) = CFAIL (Q) + IFAIL (Q)                                 00153000
      CNADIS (Q) = CNADIS (Q) + TNADIS (Q, I)                           00153100
    5 CONTINUE                                                          00153200
C                                                                       00153300
C NOW CALCULATE TOTALS & PERCENTAGES                                    00153400
C ASSUME CHANGE WEEK HAS NOT PASSED THIS QUARTER, INIT CAP & CONV TO 'A'00153500
      $CONV = CONV (1,I)                                                00153600
      $CAP = CAP (1,I) - BCAP (1,I)                                     00153700
      TBKLOG (5, I) = TBKLOG (4, I)                                     00153800
C                                                                       00153900
C NOW CALCULATE UTILIZATIONS & PERCENTAGES                              00154000
      DO 12 Q = 1, 5                                                    00154100
      BKL (Q) = 0                                                       00154200
      IF (TBKLOG (Q,I) .EQ. 0) GO TO 12                                 00154300
C                                                                       00154400
C TEST IF THERE IS A CHANGE WEEK                                        00154500
      IF (CHGWK (1,I)) 7, 7, 6                                          00154600
C                                                                       00154700
C HERE IF THERE IS A CHANGE WEEK, CHECK IF IT HAS PASSED                00154800
    6 ITEM = Q                                                          00154900
      IF (ITEM .GT. 4) ITEM = 4                                         00155000
      ITEM = (13 * ITEM) + (52 * (YEARUN - 1))                          00155100
      IF (CHGWK (1, I) .LT. ITEM) GO TO 7                               00155200
C                                                                       00155300
C CHANGEWK HAS PASSED - USE 'B' CONVEN & CAPACITY                       00155400
      $CONV = CONV (2,I)                                                00155500
      $CAP = CAP (2,I) - BCAP (2,I)                                     00155600
C                                                                       00155700
C                                                                       00155800
    7 IF ($CONV) 81112, 81112, 10                                       00155900
81112 TBKLOG (Q,I) = 0                                                  00156000
      IF (Q .EQ. 4) TBKLOG (5,I) = 0                                    00156100
      GO TO 12                                                          00156200
   10 IF ($CAP) 12, 12, 11                                              00156300
   11 TEM = $CAP                                                        00156400
      BKL (Q) = (50. /$CONV) * (TBKLOG (Q, I) / TEM)                    00156500
```

60,

TABLE V-15 (CONTINUED)

```
CARD NO      ****                    CONTENTS                    ****

1566              IF (BKL (Q) .GT. 30) BKL (Q) = 30                 00156600
1567              SBKLOG (Q) = SBKLOG (Q) + TBKLOG (Q,I)            00156700
1568              CBKLOG (Q) = CBKLOG (Q) + TBKLOG (Q,I)            00156800
1569              SBKL (Q) = SBKL (Q) + (BKL (Q) * TBKLOG (Q,I))    00156900
1570              CBKL (Q) = CBKL (Q) + (BKL (Q) * TBKLOG (Q,I))    00157000
1571           12 CONTINUE                                          00157100
1572              IF (BR1 .EQ. 0) GO TO 44444                       00157200
1573        C                                                       00157300
1574              DO 20 Q = 1,5                                     00157400
1575              TEM = AOBCON                                      00157500
1576              AOB (Q) = TSDAYS (Q,I) / TEM                      00157600
1577        C MULT BY 4 IF QUARTERLY TOTAL                          00157700
1578              IF (Q .LE. 4) AOB (Q) = AOB (Q) * 4.              00157800
1579              PASS (Q) = TUTIL (Q,I) - IFAIL (Q) - INADIS (Q)   00157900
1580              IF (PASS (Q) .LT. 0) PASS (Q) = 0                 00158000
1581              PCTUTL (Q) = 0.                                   00158100
1582              TEM = TCAP (Q, I)                                 00158200
1583              IF (TCAP (Q, I) .GT. 0)                           00158300
1584            1 PCTUTL (Q) = (TUTIL (Q, I) / TEM) * 100.          00158400
1585        C                                                       00158500
1586        C TEST UTILIZATION TO SEE IF WE CAN CALCULATE PERCENTAGES 00158600
1587              IF (TUTIL (Q, I)) 13, 13, 14                      00158700
1588        C                                                       00158800
1589        C NO UTILIZATION - ZERO PERCENTAGES & EXIT              00158900
1590           13 CONTINUE                                          00159000
1591              PCTNSH (Q) = 0.                                   00159100
1592              PCTPAS (Q) = 0.                                   00159200
1593              PCTFL (Q) = 0.                                    00159300
1594              PCTSB (Q) = 0.                                    00159400
1595              PCTNDS (Q) = 0.                                   00159500
1596              GO TO 20                                          00159600
1597        C                                                       00159700
1598        C HAVE UTILIZATION - CALC %                             00159800
1599           14 TEM = TUTIL (Q, I)                                00159900
1600              TEM1 = TNOSHO (Q, I) + TUTIL (Q, I)               00160000
1601              PCTNSH (Q) = (TNOSHO (Q, I) / TEM1) * 100.        00160100
1602              PCTPAS (Q) = (PASS (Q) / TEM) * 100.              00160200
1603              PCTFL (Q) = (IFAIL (Q) / TEM) * 100.              00160300
1604              PCTSB (Q) = (TSETBK (Q, I) / TEM) * 100.          00160400
1605              PCTNDS (Q) = (INADIS (Q) / TEM) * 100.            00160500
1606           20 CONTINUE                                          00160600
1607        C                                                       00160700
1608        C                                                       00160800
1609        C CONVERT LENGTH TO DAYS FOR PRINTOUT                   00160900
1610              DO 27 Q = 1, 3                                    00161000
1611              IF (LEN (Q,I)) 28, 28, 26                         00161100
1612           28 PLEN (Q) = 0.                                     00161200
1613              GO TO 27                                          00161300
1614           26 ITEM = LEN (Q,I) - 1                              00161400
1615              PLEN (Q) =(LEN (Q,I) * 5.) + (ITEM * 2) + .01     00161500
1616           27 CONTINUE                                          00161600
1617        C                                                       00161700
1618        C CALL OUTPUT ROUTINES, CRT IF FLAG 0 OR 1, PRINT IF 1 OR 2 00161800
1619        C SET X TO 1 FOR COURSE OUTPUT                          00161900
1620        C                                                       00162000
1621              X = 1                                             00162100
1622              IF (BR1 .EQ. 0) GO TO 12341                       00162200
1623              IF (YRFLG .LE. 1) CALL CRTOUT (X)                 00162300
1624              IF (YRFLG .GE. 1) CALL NPRINT (X)                 00162400
1625        12341 CONTINUE                                          00162500
1626        C                                                       00162600
1627        C                                                       00162700
1628        C IF I IS EQ CRSES THIS IS LAST COURSE, SO FALL THROUGH & PRINT SCHOOL 00162800
1629        C SUMMARY.  OTHERWISE GO TO END OF DO LOOP & CONTINUE.  00162900
1630        44444 IF (I .NE. CRSES) GO TO 4444                      00163000
1631        C                                                       00163100
1632        C ENTRY SCHOOL TOTALS CALC ROUTINE, SET X TO 2 FOR SCHOOL OUTPUT 00163200
1633        C CALCULATE SCHOOL TOTALS (SAME ROUTINE USED FOR CENTER) 00163300
1634        C                                                       00163400
1635          500 X = 2                                             00163500
1636        C                                                       00163600
1637        C 560 IS ENTRY FOR CENTER TOTALS CALCULATIONS           00163700
1638          560 CONTINUE                                          00163800
1639              IF (X .EQ. 2 .AND. BR2 .EQ. 0) GO TO 12342        00163900
1640              IF (X .EQ. 3 .AND. BR3 .EQ. 0) GO TO 12342        00164000
1641        C                                                       00164100
1642        C CALCULATE YEARLY TOTALS                               00164200
1643              DO 540 Q = 1, 4                                   00164300
1644              SCONV (5) = SCONV (5) + SCONV (Q)                 00164400
1645              SCAP (5) = SCAP (5) + SCAP (Q)                    00164500
1646              SUTIL (5) = SUTIL (5) + SUTIL (Q)                 00164600
1647              SNOSHO (5) = SNOSHO (5) + SNOSHO (Q)              00164700
1648              SSDAYS (5) = SSDAYS (5) + SSDAYS (Q)              00164800
1649              SFAIL (5) = SFAIL (5) + SFAIL (Q)                 00164900
1650              SSETBK (5) = SSETBK (5) + SSETBK (Q)              00165000
1651          540 SNADIS (5) = SNADIS (5) + SNADIS (Q)              00165100
1652        C                                                       00165200
1653        C                                                       00165300
1654        C NOW CALC PERCENTAGES                                  00165400
1655              DO 550 Q = 1, 5                                   00165500
1656        C                                                       00165600
1657              TEM = AOBCON                                      00165700
1658              AOB (Q) = SSDAYS (Q) / TEM                        00165800
1659              IF (Q .LE. 4) AOB (Q) = AOB (Q) * 4.              00165900
1660        C                                                       00166000
1661              PASS (Q) = SUTIL (Q) - SFAIL (Q) - SNADIS (Q)     00166100
1662              IF (PASS (Q) .LT. 0) PASS (Q) = 0                 00166200
1663        C                                                       00166300
1664              PCTUTL (Q) = 0.                                   00166400
1665              TEM = SCAP (Q)                                    00166500
1666              IF (TEM .GT. 0.) PCTUTL (Q) =(SUTIL (Q) / TEM) * 100. 00166600
1667        C                                                       00166700
1668              IF (SUTIL (Q)) 513, 513, 514                      00166800
1669          513 CONTINUE                                          00166900
1670              PCTNSH (Q) = 0.                                   00167000
1671              PCTPAS (Q) = 0.                                   00167100
1672              PCTF[illegible] (Q) = 0.                          00167200
1673              PCT[illegible] (Q) = 0.                           00167300
1674              PCTNDS (Q) = 0.                                   00167400
1675              GO TO 520                                         00167500
1676        C                                                       00167600
1677          514 TEM = SUTIL (Q)                                   00167700
```

TABLE V-15 (CONTINUED)

08/15/75 INPUT LISTING AUTOFLOW CHART SET - FINAL -KEND- FLOWCHARTS

CARD NO **** CONTENTS ****

```
      TEM1 = SNOSHO (Q) + SUTIL (Q)
      PCTNSH (Q) = (SNOSHO (Q) / TEM1) * 100.
      PCTPAS (Q) =(PASS (Q) / TEM) * 100.
      PCTFL (Q) = (SFAIL (Q) / TEM) * 100.
      PCTSB (Q) = (SSETBK (Q) / TEM) * 100.
      PCTNDS (Q) = (SNADIS (Q) / TEM) * 100.
  520 CONTINUE
      SBKL (Q) = SBKL (Q) / SBKLOG (Q)
  550 CONTINUE
C
C CALL OUTPUT SUBROUTINE, X IS 2 FOR SCHOOL OUT, 3 FOR CENTER OUT
C
      IF (YRFLG .LE. 1) CALL CRTOUT (X)
      IF (YRFLG .GE. 1) CALL NPRINT (X)
12342 CONTINUE
C
C TEST IF END OF FILE - CALC YEAR CENTER TOTALS
C TEST IF END OF FILE - GO PRINT CENTER TOTALS
      IF (ITEMS .EQ. CRSES) GO TO 580
      IF (ITEMS .GT. 900) GO TO 4444
C
C
C KEEP RUNNING - ZERO SCHOOL TOTALS & RETURN
      DO 563 Q = 1, 5
      SUTIL (Q) = 0
      SCAP  (Q) = 0
      SSETBK(Q) = 0
      SSDAYS(Q) = 0
      SNOSHO(Q) = 0
      SCONV (Q) = 0
      SFAIL (Q) = 0
      SBKL (Q) = 0
      SBKLOG (Q) = 0
  563 SNADIS(Q) = 0
C
      GO TO 4
C
C LOAD CENTER TOTALS INTO SCHOOL STORAGE - USE SCHOOL ROUTINE TO CALC
C CENTER TOTALS.
C
  580 DO 581 Q = 1, 5
      SUTIL (Q) = CUTIL (Q)
      SCAP (Q) = CCAP (Q)
      SSETBK (Q) = CSETBK (Q)
      SSDAYS (Q) = CSDAYS (Q)
      SNOSHO (Q) = CNOSHO (Q)
      SCONV (Q) = CCONV (Q)
      SFAIL (Q) = CFAIL (Q)
      SBKL (Q) = CBKL (Q)
      SBKLOG (Q) = CBKLOG (Q)
  581 SNADIS (Q) = CNADIS (Q)
C
C SET X TO 3 TO CALL CENTER OUTPUT, SET ITEMS TO 999 TO INDICATE END
      ITEMS = 999
      X = 3
      GO TO 560
C
 4444 CONTINUE
29999 CONTINUE
C EXIT IF YEARUN EQ Y AS LAST YEAR OF RUN
      IF (YEARUN .EQ. Y) GO TO 6999
      YEAR = YEAR + 1
C
C SET TREND DEMANDS
      IF (TDMAND .LE. 0) GO TO 5555
      DO 5554 ITEM = 1, CRSES
      DO 5554 ITEM1 = 1, 3
      DEMAND(ITEM1, ITEM) = DEMAND (ITEM1, ITEM) * TDMAND
 5554 BDMAND (ITEM1, ITEM) = BDMAND (ITEM1, ITEM) * TDMAND
 5555 CONTINUE
C
C ZERO TOTALS FOR MULTI YEAR RUN
      DO 6998 I = 1, CRSES
      DO 6998 Q = 1, 4
      TUTIL (Q,I) = 0
      TCAP (Q,I) = 0
      TSETBK (Q,I) = 0
      TSDAYS (Q,I) = 0
      TNOSHO (Q,I) = 0
      TCONV (Q,I) = 0
      TFAIL (Q,I) = 0
 6998 TNADIS (Q,I) = 0
 6999 CONTINUE
      RETURN
      END

      SUBROUTINE NPRINT (X)
C
C SUBROUTINE TO PRINT COURSE, SCHOOL, AND CENTER TOTALS
C THIS SUBROUTINE PRINTS FORMATTED OUTPUT AT A RATE OF 3 COURSES PER
C PAGE.  IT ALSO PRINTS TOTALS FOR A SCHOOL WHENEVER A NEW SCHOOL CODE
C IS ENCOUNTERED IN THE DATA STREAM.  IT ALSO PRINTS CENTER TOTALS AT
C THE END OF A YEAR.  SEPARATE OUTPUT FORMATS ARE PROVIDED FOR EACH YEAR
C WHENEVER THE MODEL INVOLVES MULTIPLE YEAR RUNS.
C
C
      IMPLICIT INTEGER*2 (A-Z,$)
C
C COMMON FOR CLASS DATA MATRIX
C
      REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
      REAL*4 TDMAND, SETBAK
C
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK
C
      INTEGER*4 JNO, PNEC, SCHOOL
      REAL*4 PAOB, PFAIL
```


TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
      REAL*8 NAME1, NAME2, CATNO1
C
      COMMON /TEXT/ JNO (176), NAME1 (176), NAME2 (176), CATNO1 (176)
      COMMON /TEXT/  CATNO2 (176), PNEC (176), TYPE (176), PAOR (176)
      COMMON /TEXT/ PFAIL (176), PRTY (176), SCHOOL (176)
C
      REAL*4 TFAIL, TNADIS
      INTEGER*4 TSDAYS, TCAP, TUTIL
C
      COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176)
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
      COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
C
C COMMON FOR SCHOOL AND CENTER TOTALS
C
      INTEGER*4 SUTIL, CUTIL, SCAP, CCAP, SSDAYS, CSDAYS, IFAIL, INADIS
      INTEGER*4 PASS, SBKLOG, CBKLOG, SBKL,CBKL
      REAL*4 AOB, PCTUTL, PCTNSH, PCTPAS, PCTFL, PCTSB, PCTNDS, PLEN
C
      COMMON /T/ SUTIL (5), SCAP (5), SSDAYS (5), SSETBK (5), SNOSHO (5)
      COMMON /T/ CUTIL (5), CCAP (5), CSDAYS (5), CSETBK (5), CNOSHO (5)
      COMMON /T/SBKL(5),SBKLOG(5),SCONV (5), SFAIL (5), SNADIS (5)
      COMMON /T/ CCONV(5),CBKL(5),CBKLOG(5), CFAIL (5), CNADIS (5)
      COMMON /T/ PCTUTL (5), PCTNSH (5), PCTNDS (5), INADIS (5)
      COMMON /T/ AOB (5), PASS (5), PCTPAS (5), PCTFL (5), PCTSB (5)
      COMMON /T/ IFAIL (5), PLEN (3), BKL (5), SPLAT (4), YEAR2D
C
C COMMON FOR COMMUNICATION BETWEEN SUBROUTINES
C
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN
C
      COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE
      COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
      COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
      COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON
      COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
C
      DATA PAGE /1/, PAGER /4/
       IF (PRINTF .GT. X) RETURN
C
      I = ITEM5
C
C X IS 1 FOR COURSE PRINT, 2 FOR SCHOOL PRINT, & 3 FOR CENTER PRINT
      IF (X .EQ. 3) GO TO 900
      IF (X .EQ. 2) GO TO 500
C
C BEGIN INDIVIDUAL COURSE PRINT - CHECK IF WE HAVE PRINTED 3 ON THIS PAG
C
      IF (PAGER .LE. 3) GO TO 4445
      WRITE (6,1000) YEAR,PAGE
C
      WRITE (6, 1001) SCHOOL (I)
C
      PAGER = 1
      PAGE = PAGE + 1
 4445 CONTINUE
C
      PAGER = PAGER + 1
C
C
 4999 CONTINUE
C
C NOW PRINT OUT INDIVIDUAL COURSE SUMMARY
C
      WRITE (6, 2000) COP (I), NAME1 (I), NAME2 (I)
      WRITE (6,2001)
      WRITE (6,2002) YEAR2D,YEAR2D,YEAR2D,YEAR2D
C
      WRITE (6, 2003) JNO (I), CATNO1 (I), CATNO2 (I), PNEC (I),
     1 TYPE (I), PRTY (I)
C
      WRITE (6, 2004) TCONV (1,I), TCONV (2,I), TCONV (3,I), TCONV (4,I)
     1 , TCONV (5,I)
C
      WRITE (6, 2005) TCAP (1,I), TCAP (2,I), TCAP (3,I), TCAP (4,I),
     1 TCAP (5,I)
C
      WRITE (6, 2006) PAOB (I), PFAIL (I), BKLG(I), BAKLOG (I),
     * (TUTIL (Q,I), PCTUTL (Q), Q = 1, 5)
C
      WRITE (6, 2007) SPLAT (1), SPLAT (2), SPLAT (3),
     1 (TNOSHO (Q,I), PCTNSH (Q), Q = 1, 5)
C
      WRITE (6, 2008) (TBKLOG (Q,I), BKL (Q), Q = 1, 5)
C
      WRITE (6, 2009) CONV (1,I), CONV (2,I), CONV (3,I),
     * (TSDAYS (Q,I), AOB (Q), Q = 1, 5)
C
      WRITE (6, 2010) PLEN (1), PLEN (2), PLEN (3)
C
      WRITE (6, 2011) CAP (1,I), CAP (2,I), CAP (3,I),
     * (PASS (Q), PCTPAS (Q), Q = 1, 5)
C
      WRITE (6, 2012) BCAP (1,I), BCAP (2,I), BCAP (3,I),
     * (IFAIL (Q), PCTFL (Q), Q = 1, 5)
C
      WRITE (6, 2013) DEMAND (1,I), DEMAND (2,I), DEMAND (3,I),
     * (TSETBK (Q,I), PCTSB (Q), Q = 1, 5)
C
      WRITE (6, 2014) BDMAND (1,I), BDMAND (2,I), BDMAND (3,I),
     * (INADIS (Q), PCTNDS (Q), Q = 1, 5)
C
      WRITE (6, 2015) OFFSET (I), CHGWK (1,I), CHGWK (2,I)
C
      GO TO 4444
C
C
C SCHOOL TOTALS ROUTINE
C
  500 WRITE (6, 1000) YEAR, PAGE
      J = 1
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
      PAGE = PAGE + 1                                                   00190200
      PAGER = 4                                                         00190300
C                                                                       00190400
      WRITE (6, 1001) SCHOOL (J)                                        00190500
      WRITE (6, 7001) SCHOOL (J)                                        00190600
C                                                                       00190700
C                                                                       00190800
C ENTRY FOR FTC TOTALS                                                  00190900
C FROM HERE ON SAME ROUTINE IS USED FOR SCHOOL AND CENTER TOTALS PRINT  00191000
C ROUTINE.  YEAR DATA HAS BEEN OVERWRITTEN INTO SCHOOL STORAGE AREAS.   00191100
C                                                                       00191200
  560 CONTINUE                                                          00191300
C                                                                       00191400
C                                                                       00191500
C                                                                       00191600
C PRINT ROUTINE                                                         00191700
C PRINT SCHOOL AND CENTER TOTALS                                        00191800
C                                                                       00191900
      WRITE (6, 7002)                                                   00192000
      WRITE (6, 7003) YEAR2D, YEAR2D, YEAR2D, YEAR2D                    00192100
      WRITE (6, 7004)                                                   00192200
      WRITE (6, 7005) (SCONV (Q), Q = 1, 5)                             00192300
      WRITE (6, 7006) (SCAP (Q), Q = 1, 5)                              00192400
      WRITE (6, 7007) ((SUTIL (Q), PCTUTL (Q)), Q = 1, 5)               00192500
      WRITE (6, 7008) ((SNOSHO (Q), PCTNSH (Q)), Q = 1, 5)              00192600
      WRITE (6, 7009) ((SSDAYS (Q), ADB (Q)), Q = 1, 5)                 00192700
      WRITE (6, 7010)                                                   00192800
      WRITE (6, 7011) ((PASS (Q), PCTPAS (Q)), Q = 1, 5)                00192900
      WRITE (6, 7012)((SFAIL (Q), PCTFL (Q)), Q = 1, 5)                 00193000
      WRITE (6, 7013) ((SSETBK (Q), PCTSB (Q)), Q = 1, 5)               00193100
      WRITE (6, 7014)((SNADIS (Q), PCTNDS (Q)), Q = 1, 5)               00193200
      WRITE (6, 7015)                                                   00193300
C                                                                       00193400
      GO TO 4444                                                        00193500
C                                                                       00193600
C                                                                       00193700
C HEADER FOR CENTER ANNUAL TOTALS                                       00193800
C                                                                       00193900
  960 WRITE (6, 1000) YEAR, PAGE                                        00194000
      WRITE (6, 8001)                                                   00194100
 8001 FORMAT (1H0, T42, '******** TOTALS FOR FLEET TRAINING CENTER *****00194200
     1***', /////)                                                      00194300
      GO TO 560                                                         00194400
C                                                                       00194500
 4444 CONTINUE                                                          00194600
C                                                                       00194700
C                                                                       00194800
C                                                                       00194900
C OUTPUT FORMATS FOR COURSE, SCHOOL & CENTER PRINTOUTS                  00195000
C                                                                       00195100
 1000 FORMAT (1H1,T49,'TPF MODEL -- FISCAL YEAR ',I4,' SUMMARY',T123,   00195200
     * 'PAGE ',I3)                                                      00195300
C                                                                       00195400
 1001 FORMAT (' ', T58, 'SCHOOL CODE  ', A4)                            00195500
C                                                                       00195600
 2000 FORMAT (// ' ', A4, 5X, 2A8)                                      00195700
C                                                                       00195800
 2001 FORMAT (' ', T78, 'QUARTERLY PROJECTIONS')                        00195900
C                                                                       00196000
 2002 FORMAT (' JMO   CATALOG NO   NEC   TYPE  PRTY', T60,'| 1 QTR FY ',00196100
     * I2,' | 2 QTR FY ',I2,' | 3 QTR FY ',I2,' | 4 QTR FY ',I2,        00196200
     * ' | ANNUAL TOTAL |')                                             00196300
C                                                                       00196400
 2003 FORMAT (1X, A4, 2X, A8, A2, 2X, A4, 4X, A2, 4X, A2,               00196500
     1                         T42,   '-------------------|-----------00196600
     *--|-------------|-------------|-------------|-------------|')     00196700
C                                                                       00196800
 2004 FORMAT (1X, T42, 'NUMBER CONVENINGS |', 4 (I5, ' |', 6X, '|'),    00196900
     * I6, ' |', 6X, '|')                                               00197000
C                                                                       00197100
 2005 FORMAT (' P ASS  PFAIL   BACKLOG  STU  LENGTH',                   00197200
     *              T42, 'COURSE CAPACITY   |', 4 (I5, ' |', 6X, '|'),  00197300
     * I6, ' |', 6X, '|')                                               00197400
C                                                                       00197500
 2006 FORMAT (1X, F5.1, F6.1, '%    INIT', 2X, I5, I4, ' WKS',          00197600
     *              T42, 'UTILIZATION       |', 4 (I5, ' |', F5.1, '%|'), 00197700
     * I6, ' |', F5.1, '%|')                                            00197800
C                                                                       00197900
 2007 FORMAT (1X, T19, 3 (4X, A2),                                      00198000
     *              T42, 'NO SHOWS          |', 4 (I5, ' |', F5.1, '%|'), 00198100
     * I6, ' |', F5.1, '%|')                                            00198200
C                                                                       00198300
 2008 FORMAT (' --SCHEDULE DATA--    FY  FY+1  FY+2',                   00198400
     *              T42, 'BKLOG-STUDENTS/WKS|', 4 (I5, ' |', I2, ' WKS|'),00198500
     * I6, ' |', I2, ' WKS|')                                           00198600
C                                                                       00198700
 2009 FORMAT (' ANNUAL CONVENINGS', 3I6,                                00198800
     *              T42, 'STUDENT DAYS / ADB|', 4 (I5, ' |', F5.1, ' |'), 00198900
     * I6, ' |', F5.1, ' |')                                            00199000
C                                                                       00199100
 2010 FORMAT (' COURSE LENGTH DYS', 3F6.1,                              00199200
     * T60,'|', T67,'|', T74,'|', T81,'|', T88,'|', T95,'|', T102,'|',  00199300
     * T109,'|', T116,'|', T124,'|', T131,'|')                          00199400
C                                                                       00199500
 2011 FORMAT (' COURSE CAPACITY  ', 3I6,                                00199600
     *              T42, 'TOTAL PASS        |', 4 (I5, ' |', F5.1, '%|'), 00199700
     * I6, ' |', F5.1, '%|')                                            00199800
C                                                                       00199900
 2012 FORMAT (' BUPERS SEATS     ', 3I6,                                00200000
     *              T42, 'TOTAL FAIL        |', 4 (I5, ' |', F5.1, '%|'), 00200100
     * I6, ' |', F5.1, '%|')                                            00200200
C                                                                       00200300
 2013 FORMAT (' ANNUAL DEMAND    ', 3I6,                                00200400
     *              T42, 'ACADEMIC SETBACK  |', 4 (I5, ' |', F5.1, '%|'), 00200500
     * I6, ' |', F5.1, '%|')                                            00200600
C                                                                       00200700
 2014 FORMAT (' BUPERS DEMAND    ', 3I6,                                00200800
     *              T42, 'NON-ACAD DISENROLL|', 4 (I5, ' |', F5.1, '%|'), 00200900
     * I6, ' |', F5.1, '%|')                                            00201000
C                                                                       00201100
 2015 FORMAT (' OFFSET / CHG WEEK', 3I6,                                00201200
     *              T42, '------------------------------------------------00201300
```

TABLE V-15 (CONTINUED)

```
     *------------------------------------------* ,//)
C
C
 7001 FORMAT (//////1X,T17,'SCHOOL CODE ',A4,' SUMMARY')
 7002 FORMAT (//1X,T59,'QUARTERLY PROJECTIONS')
C
 7003 FORMAT (1X,T37,'|  1 QTR FY ',I2,'  |  2 QTR FY ',I2,'  |  3 QTR
     *FY ',I2, ' |  4 QTR FY ',I2,'  |  ANNUAL TOTAL  |')
C
 7004 FORMAT (1X,T17,'--------------------|---------------|-------------
     *--|---------------|---------------|---------------|')
C
 7005 FORMAT (1X,T17,'NUMBER OF CONVENINGS|',I6,' |',7X,'|',I6,' |',7X,
     *'|',I6,' |',7X,'|',I6,' |',7X,'|',I7,' |',7X,'|')
C
 7006 FORMAT (1X,T17,'COURSE CAPACITY',5X,'|',I6,' |',7X,'|',I6,' |',
     * 7X,'|',I6,' |',7X,'|',I6,' |',7X,'|',I7,' |',7X,'|')
C
 7007 FORMAT (1X,T17,'UTILIZATION',T37,'|',I6,' |',F6.1,'%|',I6,' |',
     * F6.1,'%|',I6,' |',F6.1,'%|',I6,' |',F6.1,'%|',I7,' |',F6.1,
     * '%|')
C
 7008 FORMAT (1X,T17,'NO SHOWS',T37,'|',I6,' | ',F5.1,'%|',I6,' | ',
     * F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',F5.1,
     * '%|')
C
 7009 FORMAT (1X,T17,'STUDENT DAYS / AOB  |',I6,' |',F6.1,' |',I6,' |',
     * F6.1,' |',I6,' |',F6.1,' |',I6,' |',F6.1,' |',I7,' |',
     * F6.1,' |')
C
 7010 FORMAT (1X,T37,'|',7X,'|',7X,'|',7X,'|',7X,'|',7X,'|',7X,'|',7X,
     *'|',7X,'|',8X,'|',7X,'|')
C
 7011 FORMAT (1X,T17,'TOTAL PASS',T37,'|',I6,' | ',F5.1,'%|',I6,' | ',
     * F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',F5.1,
     * '%|')
C
 7012 FORMAT (1X,T17,'TOTAL FAIL',T37,'|',I6,' | ',F5.1,'%|',I6,' | ',
     * F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',F5.1,
     * '%|')
 7013 FORMAT(1X,T17,'ACADEMIC SETBACK    |',I6,' | ',F5.1,'%|',I6,
     * ' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',
     * F5.1,'%|')
C
 7014 FORMAT (1X,T17,'NON-ACADEM DISENROLL|',I6,' | ',F5.1,'%|',I6,
     * ' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I6,' | ',F5.1,'%|',I7,' | ',
     * F5.1,'%|')
C
 7015 FORMAT (1X,T17,'--------------------------------------------------
     *---------------------------------------------------')
C
      RETURN
      END

      SUBROUTINE CRTOUT (X)
      IMPLICIT INTEGER*2 (A-Z,$)
C
C COMMON FOR CLASS DATA MATRIX
C
      REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
      REAL*4 TDMAND, SETBAK
C
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK
C
      INTEGER*4 JNO, PNEC, SCHOOL
      REAL*4 PAOB, PFAIL
      REAL*8 NAME1, NAME2, CATNO1
C
      COMMON /TEXT/ JNO (176), NAME1 (176), NAME2 (176), CATNO1 (176)
      COMMON /TEXT/  CATNO2 (176), PNEC (176), TYPE (176), PAOB (176)
      COMMON /TEXT/ PFAIL (176), PRTY (176), SCHOOL (176)
C
      REAL*4 TFAIL, TNADIS
      INTEGER*4 TSDAYS, TCAP, TUTIL
C
      COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),  TSETBK (5, 176)
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176)
      COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)
C
C COMMON FOR SCHOOL AND CENTER TOTALS
C
      INTEGER*4 SUTIL, CUTIL, SCAP, CCAP, SSDAYS, CSDAYS, IFAIL, INADIS
      INTEGER*4 PASS, SBKLOG, CBKLOG, SBKL,CBKL
      REAL*4 AOB, PCTUTL, PCTNSH, PCTPAS, PCTFL, PCTSB, PCTNDS, PLEN
C
      COMMON /T/ SUTIL (5), SCAP (5), SSDAYS (5), SSETBK (5), SNOSHO (5)
      COMMON /T/ CUTIL (5), CCAP (5), CSDAYS (5), CSETBK (5), CNOSHO (5)
      COMMON /T/SBKL(5),SBKLOG(5),SCONV (5), SFAIL (5), SNADIS (5)
      COMMON /T/ CCONV(5),CBKL(5),CBKLOG(5), CFAIL (5), CNADIS (5)
      COMMON /T/ PCTUTL (5), PCTNSH (5), PCTNDS (5), INADIS (5)
      COMMON /T/ AOB (5), PASS (5), PCTPAS (5), PCTFL (5), PCTSB (5)
      COMMON /T/ IFAIL (5), PLEN (3), BKL (5), SPLAT (4), YEAR2D
C
C COMMON FOR COMMUNICATION BETWEEN SUBROUTINES
C
      INTEGER*4 ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5, WHERE
      REAL*4 TEM, TEM1, TEM2, TEM3, TEM4, TEM5, $LEN
C
      COMMON DUMMY, $CONV, $LEN, $CAP, $BCAP, LAST, COURSE
      COMMON $R1,$R2,$R3,$R4, TEM, TEM1, TEM2, TEM3, TEM4, TEM5
      COMMON ITEM, ITEM1, ITEM2, ITEM3, ITEM4, ITEM5
      COMMON YEAR, CRSES, Q, W, WKOFF, INIT, WHERE, NEGFLG, AOBCON
      COMMON TRACEF, RLSFLG, ENDFLG, PREFLG, YRFLG, YEARUN,PRINTF
C
C
C
C X IS 1 FOR COURSE OUTPUT, 2 FOR SCHOOL, 3 FOR CENTER (LAST)
```

TABLE V-15 (CONTINUED)

08/15/75 INPUT LISTING AUTOFLOW CHART SET - FINAL -KEND- FLOWCHARTS

CARD NO **** CONTENTS ****

```
C ITEMS CONTAINS COURSE POINTER                                        00212600
C                                                                      00212700
       IF (PRINTF .GT.X) RETURN                                        00212800
      I = ITEMS                                                        00212900
       IF (X .EQ. 1) GO TO 1                                           00213000
C                                                                      00213100
      WRITE (11, 5001) YEAR                                            00213200
 5001 FORMAT ('1', T23, 'TPF MODEL -- FISCAL YEAR ', I4, ' SUMMARY')   00213300
C                                                                      00213400
      IF (X .EQ. 2) GO TO 5000                                         00213500
C                                                                      00213600
C CRT ROUTINE FOR CENTER TOTALS - ACTUAL DATA OUTPUT IN SCHOOL ROUTINE 00213700
      WRITE (11, 6001)                                                 00213800
 6001 FORMAT (' ', T22, '** TOTALS FOR  FLEET TRAINING CENTER **' /)   00213900
      WRITE (11, 6002)                                                 00214000
 6002 FORMAT (' FTC - NORVA  SUMMARY' /)                               00214100
      GO TO 15000                                                      00214200
C                                                                      00214300
C                                                                      00214400
C OUTPUT FORMAT  FOR CRT DISPLAY, COMPATABLE WITH OUTPUT FORMAT PROGRAM 00214500
C FORMAT - ALWA   25 LINES OF OUTPUT, 1 FOR FIRST LINE, BLANK FOR DATA 00214600
C LINES, / FU    T LINE, * FOR END OF DATA                             00214700
C                                                                      00214800
    1 WRITE (9, 4001) COP (I), NAME1 (I), NAME2 (I)                    00214900
      WRITE (9, 4002) CONV (1,I), CONV (2,I), CONV (3,I), BKLG(I)      00215000
      WRITE (9, 4003) PLEN (I), PLEN (2), PLEN (3), BAKLOG (I)         00215100
      WRITE (9, 4004) SCHOOL (I), JNO (I), CATNO1 (I), CATNO2 (I),     00215200
     * PNEC (I), CAP (1,I), CAP (2,I), CAP (3,I)                       00215300
      WRITE (9, 4005) BCAP (1,I), BCAP (2,I), BCAP (3,I), CHGWK (1,I)  00215400
      WRITE (9, 4006) DEMAND (1,I), DEMAND (2,I), DEMAND (3,I),        00215500
     * CHGWK (2,I)                                                     00215600
      WRITF (9, 4007) PAOB (I), PFAIL (I), TYPE (I), PRTY (I),         00215700
     * BDMAND (1,I), BDMAND (2,I), BDMAND (3,I), OFFSET (I)            00215800
      WRITE (9, 4008) SPLAT (1), SPLAT (2), SPLAT (3)                  00215900
      WRITE (9, 4009) YEAR2D, YEAR2D, YEAR2D, YEAR2D                   00216000
      WRITE (9, 4010)                                                  00216100
      WRITE (9, 4011) TCONV (1,I), TCONV (2,I), TCONV (3,I), TCONV (4,I)00216200
     1 , TCONV (5,I)                                                   00216300
      WRITE (9, 4012) TCAP (1,I), TCAP (2,I), TCAP (3,I), TCAP (4,I),  00216400
     1 TCAP (5,I)                                                      00216500
      WRITE (9, 4013) ((TUTIL (Q,I), PCTUTL (Q)), Q = 1, 5)            00216600
      WRITE (9, 4014) ((TNOSHO (Q,I), PCTNSH (Q)), Q = 1, 5)           00216700
      WRITE (9, 4015) ((TBKLOG (Q,I), BKL (Q)), Q = 1, 5)              00216800
      WRITE (9, 4016) ((TSDAYS (Q, I), AOB (Q)), Q = 1, 5)             00216900
      WRITE (9, 4017)                                                  00217000
      WRITE (9, 4018) ((PASS (Q), PCTPAS (Q)), Q = 1, 5)               00217100
      WRITE (9, 4019) ((IFAIL (Q), PCTFL (Q)), Q = 1, 5)               00217200
      WRITE (9, 4020) ((TSETBK (Q,I), PCTSB (Q)), Q = 1, 5)            00217300
      WRITE (9, 4021) (( INADIS (Q), PCTNDS (Q)), Q = 1, 5)            00217400
      WRITE (9, 4022)                                                  00217500
      RETURN                                                           00217600
C                                                                      00217700
C                                                                      00217800
C CRT ROUTINE FOR SCHOOL AND CENTER TOTALS, ENTRY IS 15000 FOR CENTER  00217900
C                                                                      00218000
C                                                                      00218100
 5000 WRITE (11, 5002) SCHOOL (I)                                      00218200
 5002 FORMAT (1X, T33,'SCHOOL CODE  ', A4 /)                           00218300
C                                                                      00218400
      WRITE (11, 5003) SCHOOL (I)                                      00218500
 5003 FORMAT (' ', A4, 2X, 'SCHOOL SUMMARY' /)                         00218600
C                                                                      00218700
C ENTRY CENTER TOTALS                                                  00218800
C                                                                      00218900
15000 WRITE (11, 4009) YEAR2D, YEAR2D, YEAR2D, YEAR2D                  00219000
      WRITE (11, 4010)                                                 00219100
      WRITE (11, 5004) SCONV(1), SCONV(2), SCONV(3), SCONV(4), SCONV(5) 00219200
      WRITE (11, 5005) SCAP (1), SCAP (2), SCAP (3), SCAP (4), SCAP (5) 00219300
      WRITE (11, 5006) ((SUTIL (Q), PCTUTL (Q)), Q = 1, 5)             00219400
      WRITE (11, 5007) ((SNOSHO (Q), PCTNSH (Q)), Q = 1, 5)            00219500
      WRITE (11,14015) (SBKLOG (Q),SBKL(Q),Q=1,5)                      00219600
      WRITE (11, 5008) ((SSDAYS (Q), AOB (Q)), Q = 1, 5)               00219700
      WRITE (11, 4017)                                                 00219800
      WRITE (11, 5009) ((PASS (Q), PCTPAS (Q)), Q = 1, 5)              00219900
      WRITE (11, 5010) ((SFAIL (Q), PCTFL (Q)), Q = 1, 5)              00220000
      WRITE (11, 5011) ((SSETBK (Q), PCTSB (Q)), Q = 1, 5)             00220100
      WRITE (11, 5012) ((SNADIS (Q), PCTNDS (Q)), Q = 1, 5)            00220200
      WRITE (11, 4022)                                                 00220300
      RETURN                                                           00220400
C                                                                      00220500
C                                                                      00220600
C FORMATS FOR 80 COLUMN CRT OUTPUT                                     00220700
C                                                                      00220800
 4001 FORMAT ('1', A4, 5X, 2A8, 4X, '--SCHEDULE DATA--   FY   FY+1   FY+00220900
     *2   CRS BACKLOG')                                                00221000
C                                                                      00221100
 4002 FORMAT (' ', T31, 'ANNUAL CONVENINGS', I5, 2I7, 4X, '# STU', I5) 00221200
C                                                                      00221300
 4003 FORMAT (' SCHL JNO  CATALOG NO  NEC    COURSE LENGTH DYS', F5.1, 00221400
     * 2F7.1, 4X, '# WKS', I5)                                         00221500
C                                                                      00221600
 4004 FORMAT (' ', 2 (A4, 1X), A8, A2, 1X, A4, 4X, 'COURSE CAPACITY',  00221700
     * 3I7)                                                            00221800
C                                                                      00221900
 4005 FORMAT (' ', T31, 'BUPERS CAPACITY', 3I7, 3X, 'FY+1 WK', I4)     00222000
C                                                                      00222100
 4006 FORMAT (' P AOB  PFAIL  TYPE   PRTY    ANNUAL DEMAND', 2X, 3I7,  00222200
     * 3X, 'FY+2 WK', I4)                                              00222300
C                                                                      00222400
 4007 FORMAT (' ', F5.1, F6.1, '%', 3X, 2 (A2, 5X), 'BUPERS DEMAND',   00222500
     * 2X, 3I7, 3X, 'OFFSET', I5)                                      00222600
C                                                                      00222700
 4008 FORMAT (1X, T46, 3 (5X, A2), / 1X, T33, 'QUARTERLY PROJECTIONS') 00222800
C                                                                      00222900
 4009 FORMAT (' ', 12X, '| 1 QR FY', I3, ' | 2 QR FY', I3,             00223000
     1 ' | 3 QR FY', I3, ' | 4 QR FY', I3, ' | ANNUAL TOTAL |')        00223100
C                                                                      00223200
 4010 FORMAT (' ', '------------|------------|------------|',          00223300
     1 '------------|------------|-------------|')                     00223400
C                                                                      00223500
 4011 FORMAT (' ', '# CONVENINGS|', I4, ' |', 5X,   ' |', I4, ' |',    00223600
     1 5X,   ' |', I4, ' |', 5X,  ' |', I4, ' |', 5X,   ' |',          00223700
```

TABLE V-15 (CONTINUED)

CARD NO **** CONTENTS ****

```
     2 I6, ' |', 5X,    ' |')
C
 4012 FORMAT (' ', 'CRS CAPACITY|', I4, ' |', 5X,    ' |', I4, ' |',
     1 5X,    ' |', I4, ' |', 5X,  ' |', I4, ' |', 5X,    ' |',
     2 I6, ' |', 5X,    ' |')
C
 4013 FORMAT (' ', 'UTILIZATION |', I4, ' |', F5.1, '%|', I4, ' |',
     1 F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 4014 FORMAT (' ', 'NO SHOWS    |', I4, ' |', F5.1, '%|', I4, ' |',
     1 F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 4015 FORMAT (' ', 'BAKLOG #/WKS|', I4, ' |', I2,' WKS|', I4, ' |',
     1 I2,' WKS|', I4, ' |', I2,' WKS|',I4, ' |', I2,' WKS|',
     2 I6, ' |', I2,' WKS|')
C
 4016 FORMAT (' ', 'STU DAYS/AOB|', I4, ' |', F5.1, ' |', I4, ' |',
     1 F5.1, ' |', I4, ' |', F5.1,' |', I4, ' |', F5.1, ' |',
     2 I6, ' |', F5.1, ' |')
C
 4017 FORMAT (' ', 12X, '|', 5X, '|', 6X, '|', 5X, '|', 6X, '|', 5X,
     1 '|', 6X, '|', 5X, '|', 6X, '|', 7X, '|', 6X, '|')
C
 4018 FORMAT (' ', 'TOTAL PASS  |', I4, ' |', F5.1, '%|', I4, ' |',
     1 F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 4019 FORMAT (' ', 'TOTAL FAIL  |', I4, ' |', F5.1, '%|', I4, ' |',
     1 F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 4020 FORMAT (' ', 'SETBACK ACAD|', I4, ' |', F5.1, '%|', I4, ' |',
     1 F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 4021 FORMAT (' ', 'DISENROLL NA|', I4, ' |', F5.1, '%|', I4, ' |',
     1 F5.1, '%|', I4, ' |', F5.1,'%|', I4, ' |', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 4022 FORMAT (' ', '-----------------------------------------',
     *            '-----------------------------------------',
     2 /'/'/' ')
C
C SCHOOL AND CENTER CRT FORMATS
C
 5004 FORMAT (' ', '# CONVENINGS|', I5,  '|', 5X,    ' |', I5,  '|',
     1 5X,    ' |', I5,  '|', 5X,  ' |', I5,  '|', 5X,    ' |',
     2 I6, ' |', 5X,    ' |')
C
 5005 FORMAT (' ', 'CRS CAPACITY|', I5,  '|', 5X,    ' |', I5,  '|',
     1 5X,    ' |', I5,  '|', 5X,  ' |', I5,  '|', 5X,    ' |',
     2 I6, ' |', 5X,    ' |')
C
 5006 FORMAT (' ', 'UTILIZATION |', I5,  '|', F5.1, '%|', I5,  '|',
     1 F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 5007 FORMAT (' ', 'NO SHOWS    |', I5,  '|', F5.1, '%|', I5,  '|',
     1 F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
14015 FORMAT (' ', 'BAKLOG #/WKS|', I5, '|', I2,' WKS|', I5, '|',
     1 I2,' WKS|', I5, '|', I2,' WKS|',I5, '|', I2,' WKS|',
     2 I6, ' |', I2,' WKS|')
C
 5008 FORMAT (' ', 'STU DAYS/AOB|', I5,  '|', F6.1,  '|', I5,  '|',
     1 F6.1,  '|', I5,  '|', F6.1, '|', I5,  '|', F6.1,  '|',
     2 I6, ' |', F6.1,  '|')
C
 5009 FORMAT (' ', 'TOTAL PASS  |', I5,  '|', F5.1, '%|', I5,  '|',
     1 F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 5010 FORMAT (' ', 'TOTAL FAIL  |', I5,  '|', F5.1, '%|', I5,  '|',
     1 F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 5011 FORMAT (' ', 'SETBACK ACAD|', I5,  '|', F5.1, '%|', I5,  '|',
     1 F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
 5012 FORMAT (' ', 'DISENROLL NA|', I5,  '|', F5.1, '%|', I5,  '|',
     1 F5.1, '%|', I5,  '|', F5.1,'%|', I5,  '|', F5.1, '%|',
     2 I6, ' |', F5.1, '%|')
C
      RETURN
      END

      BLOCK DATA
C
C INITIAL VALUES FOR PARAMETERS IN COMMON STORAGE
C
      IMPLICIT INTEGER*2 (A-Z,$)
      REAL*4 LEN, NOSHOW, NOSHPL, BKLGPL, BPOOL, NADIS, FAIL, CDP
      REAL*4 TDMAND, SETBAK
C
      COMMON /DATA/ CDP (176), CRSRUN (176), CONV (3,176), LEN (3,176)
      COMMON /DATA/ CAP (3,176), BCAP (3,176), CHGWK (2,176)
      COMMON /DATA/ OFFSET (176), DEMAND (3,176), BDMAND (3,176)
      COMMON /DATA/ BPOOL (176), FAIL (176), NADIS (176), NOSHPL (176)
      COMMON /DATA/ BAKLOG (176), NOSHOW (176), BKLGPL (176)
      COMMON /DATA/ BKLG (176), TDMAND, SETBAK
C
      DATA BKLG/176*0/, CRSRUN /176*0/, BKLGPL, NOSHPL, BPOOL /528*0./
C
C QUARTERLY AND ANNUAL COURSE TOTALS ACCUMULATION
      REAL*4 TFAIL, TNADIS
      INTEGER*4 TSDAYS, TCAP, TUTIL
C
      COMMON /TOTALS/ TCAP (5, 176),   TUTIL (5, 176),   TSETBK (5, 176)
```

TABLE V-15 (CONTINUED)

08/15/75 INPUT LISTING AUTOFLOW CHART SET - FINAL -KEND- FLOWCHARTS

CARD NO **** CONTENTS ****

```
      COMMON /TOTALS/ TSDAYS (5, 176), TFAIL (5, 176),  TNOSHO (5, 176) 00235000
      COMMON /TOTALS/ TNADIS (5, 176), TBKLOG (5, 176), TCONV (5, 176)  00235100
C                                                                       00235200
      DATA TCAP /880*0/, TUTIL /880*0/, TSETBK /880*0/                  00235300
      DATA TNADIS /880*0./, TBKLOG /880*0/, TCONV /880*0/               00235400
      DATA TSDAYS /880*0/, TFAIL /880*0./, TNOSHO /880*0/               00235500
C                                                                       00235600
C COMMON FOR COURSE ACTIVE MATRIX ELEMENTS                              00235700
C                                                                       00235800
      REAL*4 ACTCTR                                                     00235900
      COMMON /RUN/ ACTIVE (250), ACTCTR (250), ACTUTL (250), ACTNEG(250)00236000
C                                                                       00236100
      DATA ACTIVE /250*0/, ACTCTR /250*0./, ACTUTL, ACTNEG /500*0/      00236200
C                                                                       00236300
C COMMON FOR SCHOOL AND CENTER TOTALS                                   00236400
C                                                                       00236500
      INTEGER*4 SUTIL, CUT[illegible], SCAP, CCAP, SSDAYS, CSDAYS, IFAIL, INADIS 00236600
      INTEGER*4 PASS, SBK[illegible], CBKLOG, SBKL,CBKL                 00236700
      REAL*4 AOB, PCTUTL, PCTNSH, PCTPAS, PCTFL, PCTSB, PCTNDS, PLEN    00236800
C                                                                       00236900
      COMMON /T/ SUTIL (5), SCAP (5), SSDAYS (5), SSETBK (5), SNOSHO (5)00237000
      COMMON /T/ CUTIL (5), CCAP (5), CSDAYS (5), CSETBK (5), CNOSHO (5)00237100
      COMMON /T/SBKL(5),SBKLOG(5),SCONV (5), SFAIL (5), SNADIS (5)      00237200
      COMMON /T/ CCONV(5),CBKL(5),CBKLOG(5), CFAIL (5), CNADIS (5)      00237300
      COMMON /T/ PCTUTL (5), PCTNSH (5), PCTNDS (5), INADIS (5)         00237400
      COMMON /T/ AOB (5), PASS (5), PCTPAS (5), PCTFL (5), PCTSB (5)    00237500
      COMMON /T/ IFAIL (5), PLEN (3), BKL (5), SPLAT (4), YEAR20        00237600
C                                                                       00237700
      DATA SPLAT /' *',3*'  '/                                          00237800
C                                                                       00237900
      END                                                               00238000
```

TABLE V-15 (CONTINUED)

615

UNCLASSIFIED
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* International Business Machines Corp., Civil/Space Sys., Federal Systems Division, Cape Kennedy Facility, 7900 N. Astronaut Blvd., Cape Canaveral, Fl 32920	2a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED 2b. GROUP N/A

3. REPORT TITLE

DESIGN OF TRAINING SYSTEMS - PROGRAM MAINTENANCE MANUAL DATA BASE, ETE, SCRR, AND TPF MODELS

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

5. AUTHOR(S) *(First name, middle initial, last name)*

Harold J. Bellamy James D. Staley
Kenneth V. Branch Ronald E. Yanko

6. REPORT DATE September 1975	7a. TOTAL NO. OF PAGES 612	7b. NO. OF REFS 0
8a. CONTRACT OR GRANT NO. N61330-73-C-0097 b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) TAEG REPORT NO. 29 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*	

10. DISTRIBUTION STATEMENT

Approved for Public Release;
Distribution is Unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Training Analysis and Evaluation Group, Orlando, Fl

13. ABSTRACT

This report contains detailed information on the three Design of Training Systems (DOTS) models and the DOTS data base. It consists of a single volume containing a description and macro flow, detailed logic flows and program listings for the Educational Technology Evaluation model (ETE), the System Capabilities Requirements and Resources (SCRR) model, the Training Process Flow (TPF) model and the DOTS data base.

Control logic, input/output record formats, and temporary and permanent data files are described for each subsystem (the three models and the data base). The information contained in this volume is intended for use by programmers tasked with installing or modifying the DOTS programs.

DD FORM 1 NOV 65 1473 (PAGE 1)
S/N 0102-014-6600

616

UNCLASSIFIED
Security Classification

14. KEY WORDS	LINK A ROLE	LINK A WT	LINK B ROLE	LINK B WT	LINK C ROLE	LINK C WT
Training Management Systems						
Computer Based Models						
Training Resource Evaluation Modeling						
Interactive Training Data Base						
Educational Technology Evaluation						